AD-A088 422 PURDUE UNIV LAFAYETTE IND F/G 9/3
PRANC: PROGRAM FOR ANALYZING NONLINEAR CIRCUITS.(U)
MAY 80 H K THAPAR, B J LEON F30602-78-C-0102
UNCLASSIFIED RADC-TR-80-139 NL

1 OF 3

AD A
088422

LEVEL

RADC-TR-80-139
Phase Report
May 1980

AD A088422

PRANC: PROGRAM FOR ANALYZING NONLINEAR CIRCUITS

Purdue University

H. K. Thapar
B. J. Leon

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-80-139 has been reviewed and is approved for publication.

APPROVED: Jon B. Valente
JON B. VALENTE
Project Engineer

APPROVED: David C. Luke
DAVID C. LUKE, Lt Colonel, USAF
Chief, Reliability and Compatibility Division

FOR THE COMMANDER: John P. Huss
JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (RBCT) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-80-139	2. GOVT ACCESSION NO. AD-A088	3. RECIPIENT'S CATALOG NUMBER 422
4. TITLE (and Subtitle) PRANC: PROGRAM FOR ANALYZING NONLINEAR CIRCUITS		5. TYPE OF REPORT & PERIOD COVERED Phase Report Nov 78 - Nov 79
		6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) H. K. Thapar, B. J. Leon		8. CONTRACT OR GRANT NUMBER(s) F30602-78-C-0102
9. PERFORMING ORGANIZATION NAME AND ADDRESS Purdue University, West Lafayette IN 47907		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62702F 233803PE
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (RBCT) Griffiss AFB NY 13441		12. REPORT DATE May 1980
		13. NUMBER OF PAGES 232
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Jon B. Valente (RBCT)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Sinusoidal Steady-state; Functional Analysis; Computer-aided Analysis; Multi-dimensional Transforms

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The application of the Volterra series method for the analysis of circuits with multiple nonlinear elements and multiple multi-frequency input sources is described.

Using multi-dimensional transform theory, an approach for deriving the Volterra transfer functions is given. The computation of the sinusoidal steady-state response from these transfer functions is presented.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Finally, algorithms for adapting the Volterra series method for computer-aided steady-state analysis of nonlinear circuits are described. A complete documentation of the program PRANC, which uses the Volterra series approach, is also contained in this report.

PREFACE

This effort was conducted by Purdue University under the sponsorship of the Rome Air Development Center Post-Doctoral Program. Mr. Jon Valente of RADC was the task project engineer and provided overall technical direction and guidance. Prof. B. J. Leon directed this research and the preparation of this report at Purdue University. The authors of the report are B. J. Leon and H. K. Thapar.

The RADC Post-Doctoral Program is a cooperative venture between RADC and some sixty-five universities eligible to participate in the program. Syracuse University, Clarkson College of Technology and the Georgia Institute of Technology act as prime contractor schools with other schools participating via sub-contracts with the prime schools. The U.S. Air Force Academy (Dept. of Electrical Engineering), Air Force Institute of Technology (Dept. of Electrical Engineering) and the Naval Post Graduate School (Dept. of Electrical Engineering) also participate in the program.

The Post-Doctoral Program provides an opportunity for faculty at participating universities to spend up to one year full time on exploratory development and problem-solving efforts with the post-doctorals splitting their time between the customer location and their educational institutions.

Further information about the RADC Post-Doctoral Program can be obtained from Jacob Scherer, RADC/RBC, Griffiss AFB, NY, 13441, telephone AV 587-2543, COMM (315) 330-2543.

This document is the final report for Task 7 of Purdue University's Subcontract from Clarkson College of Technology. The task was to "Develop and Apply Symbolic Methods to the Volterra Series Approach to Nonlinear Circuit Analysis."

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1-1. Statement of the Objectives

In the analysis of nonlinear systems, two main classes of solutions are generally sought: 1) transient, and 2) steady state. The basic goal of this investigation is to obtain the sinusoidal steady-state solution of nonlinear circuits via the Volterra series method [1-14].

The most commonly used present-day approach for analyzing nonlinear systems is numerical integration [20]. The nonlinear differential equations are integrated from some initial time, t_0, to some final time, t_f. When the sinusoidal steady-state response is sought, the value of t_f chosen is usually large to insure that all transients have been eliminated. A subsequent fast Fourier analysis yields the frequency components of the output response. A more efficient method for obtaining the sinusoidal steady-state response is to pose the analysis problem as a two-point boundary value problem and then apply Newton's method [20]. This approach, however, allows for only single frequency inputs.

The problems involved in the numerical integration method are well known [20]. These problems notwithstanding, there are other inefficiencies. When one is solely interested in the steady-state response, the computation expended in reaching t_f is a waste. This inefficiency grows as the poles of the linearized system move close to the imaginary axis, as is often the case in many quasi-linear communication circuits.

Other methods such as the harmonic balance or the describing function method are seldom used, simply because the assumption behind these methods render them undependable. The Picard iteration method [14] is often used in nonlinear systems analysis. This method also has limitations when used for computer-aided analysis, particularly when multi-tone inputs are present.

The fundamental intent behind this report is to examine the computational aspect of the Volterra series when used for the steady-state analysis of circuits with multiple nonlinearities and multiple multi-tone input sources. A basic algorithm for adapting this method for computer-aided analysis is developed. Its implementation as a digital computer program, entitled PRANC (Program for Analyzing Nonlinear Circuits), is also included in this report.

1-2. Organization of the Report

After this introductory chapter, this report contains the following five chapters.

Chapter 2, entitled "Volterra Series Method", discusses the analysis method which forms the basis of this investigation. A systematic approach for system characterization in the transform domain is developed. The determination of the sinusoidal steady-state response for multi-tone inputs from the system characterization is also developed.

Chapter 3 considers the computational aspect of the Volterra series method. An algorithm, which uses semi-symbolic analysis [20], is developed for the efficient implementation of this method on a digital computer. An overview of PRANC is also presented in this chapter.

Chapter 4 provides the User's guide for PRANC. Several examples to illustrate the use of this program are included here.

Chapter 5 contains the Programmer's guide for PRANC. Each sub-program listing, together with its functions, is documented in this chapter.

Finally, Chapter 6 is reserved for some concluding remarks.

CHAPTER 2

VOLTERRA SERIES METHOD

2-1. Introduction

Nonlinear systems that admit a Volterra series description are completely characterized by their nonlinear impulse response functions or the generalized transfer functions, which are the multi-dimensional transforms of the nonlinear impulse response functions. Thus, any analysis of nonlinear systems via the Volterra series method will entail the determination of either one of these functions.

The method for determining the generalized transfer functions given in [13] will be presented here. This method relies on the application of multi-dimensional transforms to a set of differential equations. In section 2-2 the multi-dimensional transform theory is introduced, along with the application of the theory to specific examples which will be subsequently used in deriving the generalized transfer functions. In section 2-3 the generalized transfer functions for an r-th order scalar nonlinear differential equation are obtained. Section 2-4 is devoted to the determination of the nonlinear transfer functions of a general multiple-node, multiple-nonlinearity circuit with a single input. The case of multiple input sources is treated in section 2-5. Section 2-6 shows the relationship between the terms in the sinusoidal steady-state response and the generalized transfer functions.

2-2. Multi-dimensional Transforms

The Laplace transform pair of a one-dimensional function, f(t), is:

$$F(s) = \int_{-\infty}^{\infty} f(t) e^{-st}\, dt \tag{2-1}$$

and

$$f(t) = \frac{1}{(2\pi j)} \int_{\sigma - j\infty}^{\sigma + j\infty} F(s) e^{st}\, ds \tag{2-2}$$

For a multi-variable function, $f(t_1, t_2, \ldots, t_n)$, the corresponding multi-dimensional transform [15] is:

$$F(s_1, s_2, \ldots, s_n) = \underbrace{\int \cdots \int}_{n\text{-fold}} f(t_1, t_2, \ldots, t_n) \exp(-s_1 t_1 - \cdots - s_n t_n)\, dt_1 \cdots dt_n \tag{2-3}$$

and

$$f(t_1, \ldots, t_n) = \frac{1}{(2\pi j)^n} \underbrace{\int \cdots \int}_{n\text{-fold}} F(s_1, \ldots, s_n) \exp(s_1 t_1 + \cdots + s_n t_n)\, ds_1 \ldots ds_n \tag{2-4}$$

$$f(t_1, \ldots, t_n) \leftrightarrow F(s_1, \ldots, s_n) \tag{2-5}$$

Before proceeding further, we make the following notational definitions:

$$F(s_1, s_2, \ldots, s_n) = \ell[f(t_1, t_2, \ldots, t_n)] \tag{2-6}$$

and

$$f(t_1, t_2, \ldots, t_n) = \ell^{-1}[F(s_1, s_2, \ldots, s_n)] \tag{2-7}$$

Whether we use Fourier transform or Laplace transform in eqns. (2-3) and (2-4) depends on the contours of integration and values of $s_1, s_2, \ldots, s_n$. The importance of the region of convergence when dealing with unstable and non-causal linear systems is well known. Here we assume that the systems under consideration are causal; that is, the Volterra kernels

$h_n(t_1,t_2,...,t_n) = 0$, for $t_1,t_2,...,t_n \leq 0$. Also, in general, we are concerned with functions (or generalized functions) whose region of convergence includes the imaginary axis in each variable, so that the Fourier transform is included in our definitions.

It should also be noted that most of the properties of the one-dimensional transform (linear case) carry over to the multi-dimensional case. The validity of this statement can be checked elsewhere [5].

It is often desirable to express the multi-variable function, $f(t_1,t_2,...,t_n)$, as a simple function of time, $f(t)$, and vice versa. If all t_i's are restricted to be identical so that $t = t_1 = t_2 = \cdots = t_n$, then $f(t_1,t_2,...,t_n)$ becomes $f(t)$. Thus, in the two variable case, $f(t)$ can be obtained from $f(t_1,t_2)$ by evaluating $f(t_1,t_2)$ along the 45° line $t_1 = t_2$. Similarly, if we plot $f(t_1,t_2,t_3)$ in a three-dimensional space, then, to obtain $f(t)$, we are only interested in $f(t_1,t_2,t_3)$ along the line $t_1 = t_2 = t_3$. The idea of converting a nonlinear function of one variable t into a product of linear multi-variable functions will be used repeatedly in the sequel. One must, however, bear in mind that the ultimate goal is to obtain the solution of the differential equation as a function of time, t, and that the introduction of t_1,t_2, etc. are merely for mathematical manipulations.

We now apply multi-dimensional transforms to some specific cases which will be subsequently used in sections (2-3) and (2-4).

2-2.1 <u>Volterra Series</u>: The Volterra series relates the system input $x(t)$ to the system output $y(t)$ as follows*:

*Unless otherwise stated, all limits of integration are between 0 and ∞ in our discussion here.

$$y(t) = \sum_{n=1}^{\infty} \underbrace{\int \cdots \int}_{n\text{-fold}} h_n(\tau_1, \ldots, \tau_n) \prod_{i=1}^{n} x(t-\tau_i) d\tau_i$$

$$= \sum_{n=1}^{\infty} y_n(t) \tag{2-8}$$

where

$$y_n(t) = \underbrace{\int \cdots \int}_{n\text{-fold}} h_n(\tau_1, \ldots, \tau_n) \prod_{i=1}^{n} x(t-\tau_i) d\tau_i \tag{2-9}$$

Introducing dummy variables $t_1, t_2, \ldots, t_n$ in eqn. (2-9) we can write $y_n(t)$ as:

$$y_n(t) = y_n(t_1, t_2, \ldots, t_n)\big|_{t_1=t_2=\ldots=t_n=t}$$

$$= \underbrace{\int \cdots \int}_{n\text{-fold}} h_n(\tau_1, \ldots, \tau_n) \prod_{i=1}^{n} x(t_i-\tau_i) d\tau_i \tag{2-10}$$

Taking the n-dimensional transforms of eqn. (2-10), we get:

$$Y_n(s_1, \ldots, s_n) = \mathcal{L}[y_n(t_1, \ldots, t_n)]$$

$$= \underbrace{\int \cdots \int}_{2n\text{-fold}} h_n(\tau_1, \tau_2, \ldots, \tau_n) \prod_{i=1}^{n} x(t_i-\tau_i) e^{-s_i t_i} d\tau_i dt_i \tag{2-11}$$

Defining $t_n - \tau_n = \sigma_n$, $t_{n-1} - \tau_{n-1} = \sigma_{n-1}, \ldots, t_1 - \tau_1 = \sigma_1$, and therefore: $t_n = \sigma_n + \tau_n$, $t_{n-1} = \sigma_{n-1} + \tau_{n-1}, \ldots, t_1 = \sigma_1 + \tau_1$; $d\sigma_n = dt_n$, $d\sigma_{n-1} = dt_{n-1}, \ldots, d\sigma_1 = dt_1$. Substituting these quantities in eqn. (2-11) and performing the 2n-fold integrations with respect to τ_i and

σ_i, gives

$$Y_n(s_1,\ldots,s_n) = H_n(s_1,\ldots,s_n) \prod_{i=1}^{n} X(s_i) \qquad (2\text{-}12)$$

where $H_n(s_1,\ldots,s_n)$ and $X(s_i)$ are the transforms of $h_n(t_1,t_2,\ldots,t_n)$ and $x(t_i)$ respectively. Therefore the transform domain description of eqn. (2-8) becomes:

$$Y(s_1,s_2,\ldots,s_n) = \sum_{n=1}^{\infty} H_n(s_1,\ldots,s_n) \prod_{i=1}^{n} X(s_i) \qquad (2\text{-}13)$$

If the input x(t) is a delta function, then eqns. (2-12) and (2-13) reduce, respectively, to:

$$Y_n(s_1,\ldots,s_n) = H_n(s_1,s_2,\ldots,s_n) \qquad (2\text{-}14)$$

and

$$Y(s_1,\ldots,s_n) = \sum_{n=1}^{\infty} H_n(s_1,\ldots,s_n) \qquad (2\text{-}15)$$

Equations (2-12) through (2-14) will be used repeatedly in section (2-3).

2-2.2 Nonlinear Terms. The characteristics of nonlinear elements encountered in many nonlinear dynamical systems can be represented over any finite range by a polynomial. This gives rise to nonlinear differential equations with polynomial type nonlinear terms. When such elements are used in a system, the equilibrium equations contain integrals and derivatives of the polynomials. We can apply multi-dimensional transforms to these nonlinear terms by first converting an nth power to an n-fold product of terms with different domains. More detail on these derivatives is given in [13].

$\underline{y^n(t)}$ **Term:** Consider an n-dimensional time space with variables t_i, $i=1,2,\ldots,n$. From the single variable function $y(t)$ define an n-variable functional $y(t_1,t_2,\ldots,t_n) = \prod_{i=1}^{n} y(t_i)$. Then

$$y^n(t) = y(t_1,t_2,\ldots,t_n) \qquad \forall\ t_i = t \tag{2-16}$$

and

$$Y(s_1,\ldots,s_n) = \prod_{i=1}^{n} Y(s_i) \tag{2-17}$$

$\underline{\frac{d}{dt}\, y^n(t)}$ **Term:**

$$\frac{d}{dt}\, y^n(t) = \frac{d}{dt}\, y(t_1,\ldots,t_n)\Big|_{t_1=t_2=\ldots=t_n=t}; \tag{2-18}$$

$$Y(s_1,s_2,\ldots,s_n) = \mathcal{L}\Big[\sum_{s=1}^{n} \frac{\partial}{\partial t_s}\, y(t_1,\ldots,t_n)\, \frac{dt_s}{dt}\Big]$$

$$= (s_1 + s_2 + \ldots s_n)Y(s_1)\ldots Y(s_n) \tag{2-19}$$

$\underline{\int y^n(t)dt}$ **Term:**

$$\int y^n(t)dt = \int y_n(\tau_1-t,\tau_2-t,\ldots,\tau_n-t)dt \tag{2-20}$$

Letting $\tau_i - t = t_i$, and taking the transform of eqn. (2-20), we get

$$Y(s_1,s_2,\ldots,s_n) = Y_n(s_1,\ldots,s_n)/(s_1+s_2+\ldots+s_n) \tag{2-21}$$

$$= \left[\frac{1}{s_1+s_2+\ldots+s_n}\right]\prod_{i=1}^{n} Y(s_i) \qquad (2\text{-}22)$$

The general forms in eqns. (2-12), (2-19) and (2-27) will be used in sections (2-3) and (2-4). The salient feature in each of these equations is how an nth degree polynomial function in the time-domain is represented by the nth-order product of the transform of the function in the transform domain. It is this product structure which, analogous to the case of linear system analysis, makes the analysis of nonlinear systems easier via the transform-domain approach.

2-3. A Nonlinear Differential Equation:

In this section, we present a method, based on applying the multi-dimensional transforms to nonlinear differential equations, to determine the response of a nonlinear system with a functional power series type of nonlinearity. The nonlinear differential equation considered is the following:

$$L_1[y(t)] + L_2[\sum_{n=2}^{N} a_n y^n(t)] = x(t) \qquad (2\text{-}23)$$

where x(t) and y(t) are system input and output, respectively, L_1 is a linear differential operator:

$$L_1[\cdot] = \sum_{r=0}^{R} \frac{d^r}{dt^r}[\cdot] \qquad (2\text{-}24)$$

and L_2 is $\frac{d}{dt}$, $\int$, or a constant, or a sum of these operators. It should be noted that the linear operator, L_2, operates on a polynomial function of y(t).

We now present an approach whereby the nonlinear differential equation (2-23) is solved by a bootstrapping operation by first dissolving it into a set of linear differential equations with nonlinear inputs. Multidimensional transforms are then applied to these new equations to obtain the Volterra series solution.

There are many different methods of rendering a nonlinear differential equation into a sequence of linear differential equations involving successively higher order outputs with known nonlinear input terms. We use the approach outlined in [12].

Assume that the input in eqn. (2-23) is of the form

$$x(t) = \epsilon v(t) \tag{2-25}$$

The dummy variable ϵ helps to keep track of the order of the terms: a term with coefficient ϵ^n signifies an nth order term. This can be seen easily by substituting eqn. (2-25) in eqn. (2-9), which yields:

$$y_n(t) = \epsilon^n \underbrace{\int \ldots \int}_{n\text{-fold}} h_n(\tau_1, \ldots, \tau_n) \prod_{i=1}^{n} v(t-\tau_i) d\tau_i \tag{2-26}$$

Let us assume that r(t) is the response to the input v(t) in eqn (2-23). Then, according to the Volterra series expansion, as per eqn. (2-8) and (2-9), the n-th order response is:

$$r_n(t) = \underbrace{\int \ldots \int}_{n\text{=fold}} h_n(\tau_1, \ldots, \tau_n) \prod_{i=1}^{n} v(t-\tau_i) d\tau_i \tag{2-27}$$

Comparing (2-27) and (2-26), we obtain the following relationships:

$$y_n(t) = \epsilon^n r_n(t) \tag{2-28}$$

and therefore, as per eqn. (2-8),

$$y(t) = \sum_{n=1}^{\infty} y_n(t) = \sum_{n=1}^{\infty} \epsilon^n r_n(t) \tag{2-29}$$

We now have two differential equations which relate r(t) and v(t). First, equation (2-23) can be re-written as:

$$L_1[r(t)] + L_2[\sum_{n=2}^{N} a_n r^n(t)] = v(t) \tag{2-30}$$

Second, after substituting eqn. (2-29) into (2-23), we get:

$$L_1[\sum_{n=1}^{\infty} \epsilon^n r_n(t)] + L_2[\sum_{j=2}^{N} (a_j \sum_{n=1}^{\infty} \epsilon^n r_n(t))^j] = \epsilon v(t) \tag{2-31}$$

Thus in order to solve eqn. (2-23), we can solve eqn. (2-31) for $r_n(t)$, n = 1,2,... and substitute in eqn. (2-29) to solve for y(t) after setting $\epsilon = 1$. Setting $\epsilon = 1$ implies that x(t) = v(t), and therefore $y(t) = r(t) = \sum_n r_n(t)$. The introduction of ϵ is a mathematical artifice which helps to equate coefficients of ϵ^n on both sides of eqn. (2-31), thereby yielding linear differential equations (involving successively higher order outputs) with non-linear inputs.

To solve for $r_1(t)$, the linear system response, we equate coefficients of ϵ^1 on both sides of eqn. (2-31), thus yielding the following equation:

$$L_1[r_1(t)] = v(t) \tag{2-32}$$

Similarly we equate coefficients of ϵ^2, ϵ^3, ϵ^4, ϵ^5, and so on, on both sides of eqn. (2-31) to obtain the following equations:

$$L_1[r_2(t)] + L_2[a_2 r_1^2(t)] = 0 \tag{2-33}$$

$$L_1[r_3(t)] + L_2[2a_2 r_1(t) r_2(t) + a_3 r_1^3(t)] = 0 \tag{2-34}$$

$$L_1[r_4(t)] + L_2[a_2(2r_1(t)r_3(t) + r_2^2(t)) + 3a_3 r_1^2(t) r_2(t)$$

$$+ a_4 r_1^4(t)] = 0 \tag{2-35}$$

$$L_1[r_5(t)] + L_2[2a_2 r_1(t) r_4(t) + a_3(3r_1^2(t) r_3(t) +$$

$$+ 3r_1(t) r_2^2(t)) + 4a_4 r_1^3(t) r_2(t) + a_5 r_1^5(t)] = 0 \tag{2-36}$$

.

.

.

To solve for the generalized transfer functions of eqn. (2-30), we take the 1-dimensional transform of eqn. (2-32) and obtain:

$$L_1(s_1) R_1(s_1) = V(s_1) \tag{2-37}$$

If $v(t) = \sigma(t)$, then $V(s_1) = 1$, and therefore, according to eqn. (2-14), we have

$$R_1(s_1) = H_1(s_1) = \frac{1}{L_1(s_1)} \tag{2-38}$$

To solve for the second-order transfer functions, $H_2(s_1,s_2)$, we extend the second term of eqn. (2-33) to a two dimensional domain. Since the physical system is not defined when $t_1 \neq t_2$ we can assume that the extension of eqn. (2-33) holds for all t_1 and t_2. Transforming via eqn. (2-17) gives

$$L_1(s_1+s_2)R_2(s_1,s_2) + a_2L_2(s_1+s_2)R_1(s_1)R_1(s_2) = 0 \tag{2-39}$$

Using (2-14) and (2-38) in eqn. (2-39), we obtain

$$R_2(s_1,s_2) = H_2(s_1,s_2) = -\frac{a_2L_2(s_1+s_2)H_1(s_1)H_1(s_2)}{L_1(s_1+s_2)} \tag{2-40}$$

For R_3 and higher terms we find that the order of variables t_1, t_2, t_3 seems important. Physically this should not be. We can symmetrize by averaging. That is, we sum each of the nth order transfer function over all permutations of its arguments and divide by the number of components in the sum. We use an overbar to represent the symmetrized function.

$$L_1(s_1+s_2+s_3)R_3(s_1,s_2,s_3) + L_2(s_1+s_2)[2a_2\overline{R_1(s_1)R_2(s_2,s_3)}$$

$$+ a_3R_1(s_1)R_1(s_2)R_1(s_3)] = 0 \tag{2-41}$$

Again, using eqns. (2-38), (2-40), and (2-14), we get

$$R_3(s_1,s_2,s_3) = H_3(s_1,s_2,s_3) = - L_2(s_1+s_2+s_3)[2a_2\overline{H_1(s_1)H_2(s_2,s_3)}$$

$$+ a_3H_1(s_1)H_1(s_2)H_1(s_3)]/L_1(s_1+s_2+s_3) \tag{2-42}$$

In a similar manner, we can derive by inspection:

$$H_4(s_1,s_2,s_3,s_4) = - L_2(\sum_{i=1}^{4} s_i)[a_2(2\overline{H_1(s_1)H_3(s_2,s_3,s_4)}$$

$$+ \overline{H_2(s_1,s_2)H_2(s_3,s_4)}) + 3a_3\overline{H_1(s_1)H_1(s_2)H_2(s_3,s_4)}$$

$$+ a_4 \sum_{i=1}^{4} H_1(s_i))]/L_1(\sum_{i-1}^{4} s_i) \tag{2-43}$$

and

$$H_5(s_1,s_2,s_3,s_4,s_5) = -L_2(\sum_{i=1}^{5} s_i)[2a_2 \overline{H_1(s_1)H_4(s_2,s_3,s_4,s_5)}$$

$$+ 3a_3(\overline{H_1(s_1)H_1(s_2)H_3(s_3,s_4,s_5)}$$

$$+ \overline{H_1(s_1)H_2(s_2,s_3))H_2(s_4,s_5)}$$

$$+ 4a_4\overline{H_1(s_1)H_1(s_2)H_1(s_3)H_2(s_4,s_5)} + a_5 \sum_{i=1}^{5} H_1(s_i)]/L_1(\sum_{i=1}^{5} s_i) \tag{2-44}$$

The use of symmetric transfer functions is not merely for notational convenience, but is necessitated by the method we use for introducing the parameters $t_1, t_2, \ldots,$ before taking the transforms. Consider a third order term $v_3(t)$ formed as the product of a first order term $v_1(t)$ and a second order $v_2(t)$. On the three dimensional (t_1,t_2,t_3) we could write $v_3(t_1,t_2,t_3)$ as $v_1(t_1)v_2(t_2,t_3)$, $v_1(t_2)v_2(t_1,t_3)$, or $v_1(t_3)v_2(t_1,t_2)$. The first term has transform: $V_1(s_1)V_2(s_2,s_3)$; the second term has: $V_1(s_2)V_2(s_1,s_3)$; and the third has transform: $V_1(s_3)V_2(s_1,s_2)$. When $V_2(\cdot,\cdot)$ is not symmetrical in its arguments, each transformed quantity above will yield a different value. Thus, it becomes necessary to use symmetric transfer functions when performing numerical computations to obtain the system response. It can be shown that the response is unchanged when symmetrized transfer functions are used. Since, in the final analysis we want the value of $v_3(t_1,t_2,t_3)$ only when $t_1=t_2=t_3$, we may write

$v_1(t)v_2(t) = \frac{1}{3}[v_1(t_1)v_2(t_2,t_3) + v_1(t_2)v_2(t_1,t_3) + v_1(t_3)v_2(t_1,t_2)]$. This does not change the contribution due to $v_1(t)v_2(t)$ in the system response. In the remaining part of this report we will assume the generalized transfer functions to be symmetric in their arguments.

To conclude this sub-section, we summarize the approach for obtaining the generalized transfer functions of a nonlinear system and also comment on the important ramification of the method. By introducing a dummy variable in the nonlinear differential equation characterizing the system, a set of differential equations of the following form was obtained:

$$L[r_n(t)] + f(\underline{r}_{n-1}(t)) = 0, \quad n = 2,3,... \tag{2-45}$$

where L is the linear system operator and f(•) is a nonlinear function of $r_{n-1}(t)$, $r_{n-2}(t)$,...,$r_1(t)$. $r_1(t)$ is the first-order response, which is simply the response of the linear system. The relationship in eqn. (2-45) is clearly a recursive one, and can be used to solve for $r_n(t)$ in terms of $r_{n-1}(t)$, $r_{n-2}(t)$, etc. This is done by first finding the n-dimensional transform of $f(r_{n-1}(t))$ as discussed above. We then use the transform of eqn. (2-44) to solve for $R_n(s_1,...,s_n)$, the nth-order transfer function when the input v(t) is an impulse. The transform of f(•) is done by inspection with the help of the results of section (2-2). The n-dimensional transform of $L[r_n(t)]$ is shown to be $L(s_1+s_2+...+s_n)R_n(s_1,s_2,...,s_n)$. With all this information, eqn. (2-45) is easily solved for the generalized transfer functions.

2-4. Multiple-Node, Multiple-Nonlinearity Circuit Analysis

Many analysis and design problems in circuits and systems involve one or at most a few nonlinear elements in an otherwise linear time-invariant circuit or system. When a single nonlinear element is present, the differential equation (2-23) and the material of section (2-2) will be adequate for analyzing the nonlinear circuit. For, in such a case, the linear circuit can be characterized by a convolution kernel (via the Thevenin or Norton Theorems) to give the overall Volterra integral equation [14], which can also be cast in a differential equation form, similar to eqn. (2-23).

However, when multiple nonlinear elements are imbedded in an otherwise linear time-invariant circuit, the analysis entails the solution of a <u>system</u> of nonlinear differential equations. The approach developed in section (2-2) for the scalar case is still applicable, but must be extended to solve the system of nonlinear differential equations.

The number of equations to be solved depends on the number and the type of nonlinear elements considered. When only independent type nonlinear elements are considered, the number of equations is less than or equal to the number of nonlinear elements (assuming that the output is across one of the nonlinear elements; otherwise, an extra equation relating the nonlinear element voltages (currents) and the output voltage (current) is needed to solve for the output). The nonlinear differential equations in such a case is again derived by obtaining the Thevenin (Norton) equivalent circuit (for the linear part of the nonlinear circuit) at each of the ports at which the nonlinear elements are present. When dependent type nonlinear elements are also allowed, the analysis becomes more complicated; for, in such a case, the controlling variables, which may be across a linear element, must be solved for and substituted in the differential equation for the nonlinear element.

Previous works [7,10-12] for determining the generalized voltage ratio transfer functions of lumped nonlinear circuits have applied the harmonic input method, to the nodal analysis. Our discussion in this section for solving multiple-node, multiple-nonlinearity circuits will be centered around the application of multi-dimensional transforms to a cutset type analysis. Thus, we will be solving for the generalized voltage ratio transfer functions. As we proceed with our discussion, it will become apparent that a cutset analysis approach is the most natural way of solving for the generalized voltage-ratio transfer functions. We now develop the procedure.

The first step in the analysis is to represent each nonlinear element by a polynomial expansion. Thus, in the distortion analysis of transistor amplifiers [7], the exponential type controlled sources in the Ebers-Moll model are first represented by a Taylor series expansion of the function about the quiescent point, thereby yielding a polynomial in terms of the incremental variables. The types of nonlinear elements, and their series representation, that are commonly encountered are:

1. No memory, independent nonlinearity (Nonlinear Resistor)

$$i = F(v) = \sum_{j=1}^{\infty} a_j v^j \tag{2-46}$$

2. No memory, dependent nonlinearity

$$i = G(u,v) = \sum_{j=0}^{\infty}\sum_{k=0}^{\infty} a_{jk} u^j v^k \quad , \quad a_{00} = 0 \tag{2-47}$$

3. Capacitive, independent nonlinearity

$$i = \frac{d}{dt}Q(v) = \frac{d}{dt}\sum_{j=1}^{\infty} a_j v^j \tag{2-48}$$

4. Inductive, independent nonlinearity

$$i = \int_{-\infty}^{t} \phi(v)dt = \int_{-\infty}^{t}\sum_{j=1}^{\infty} a_j v^j dt \tag{2-49}$$

where

$i \equiv$ incremental current through the element

$v \equiv$ incremental controlling voltage

$u \equiv$ incremental controlling voltage

The general procedure employed to solve for the nonlinear transfer functions of a single-input, single-output nonlinear circuit using the cutset analysis approach is illustrated in Fig. 2-1 by considering each of the four nonlinear element types mentioned above.

Consider the nonlinear circuit N, shown in Fig. 2-1(a), containing a nonlinear resistor, a nonlinear dependent source, a nonlinear capacitor, and a nonlinear inductor, where each nonlinear element is voltage controlled. The procedure begins by identifying all the nonlinear elements, as shown in Fig. 2-1(b). We note that the four nonlinear elements depend on six voltages. The next step is to lump the linear parts of the nonlinear elements with the existing linear network to form the augmented linear network. The square, cubic, quartic, etc. terms of the nonlinearity are treated as nonlinear current sources, indicated by i_k^n, meaning the nth order current source at port k. Since the dependent source, $g(v_5,v_6)$, depends on voltages v_5 and v_6, we also extract these as ports. Thus, altogether we end up with an 8-port linear network, as shown in Fig. 2-1(c).

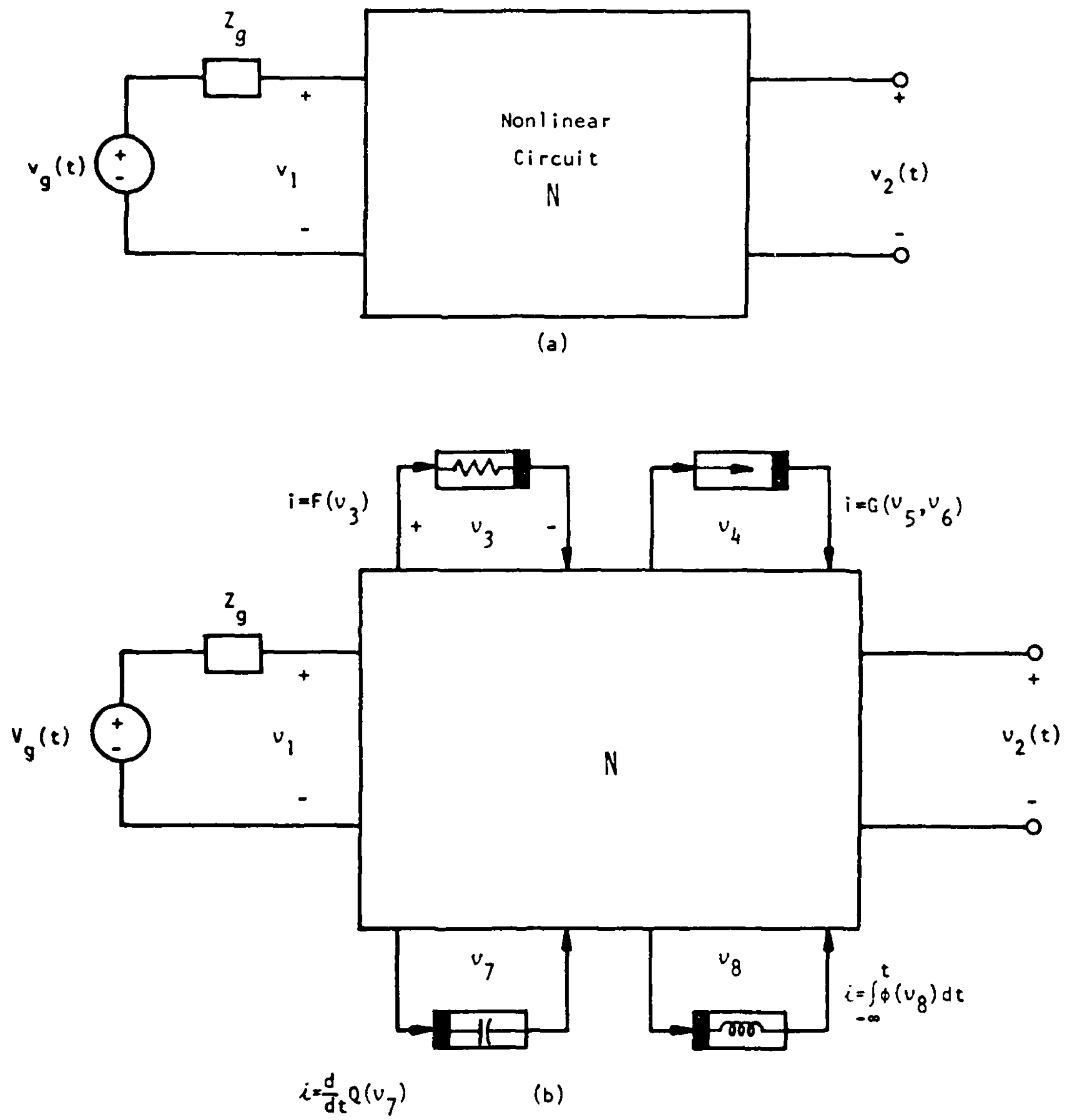
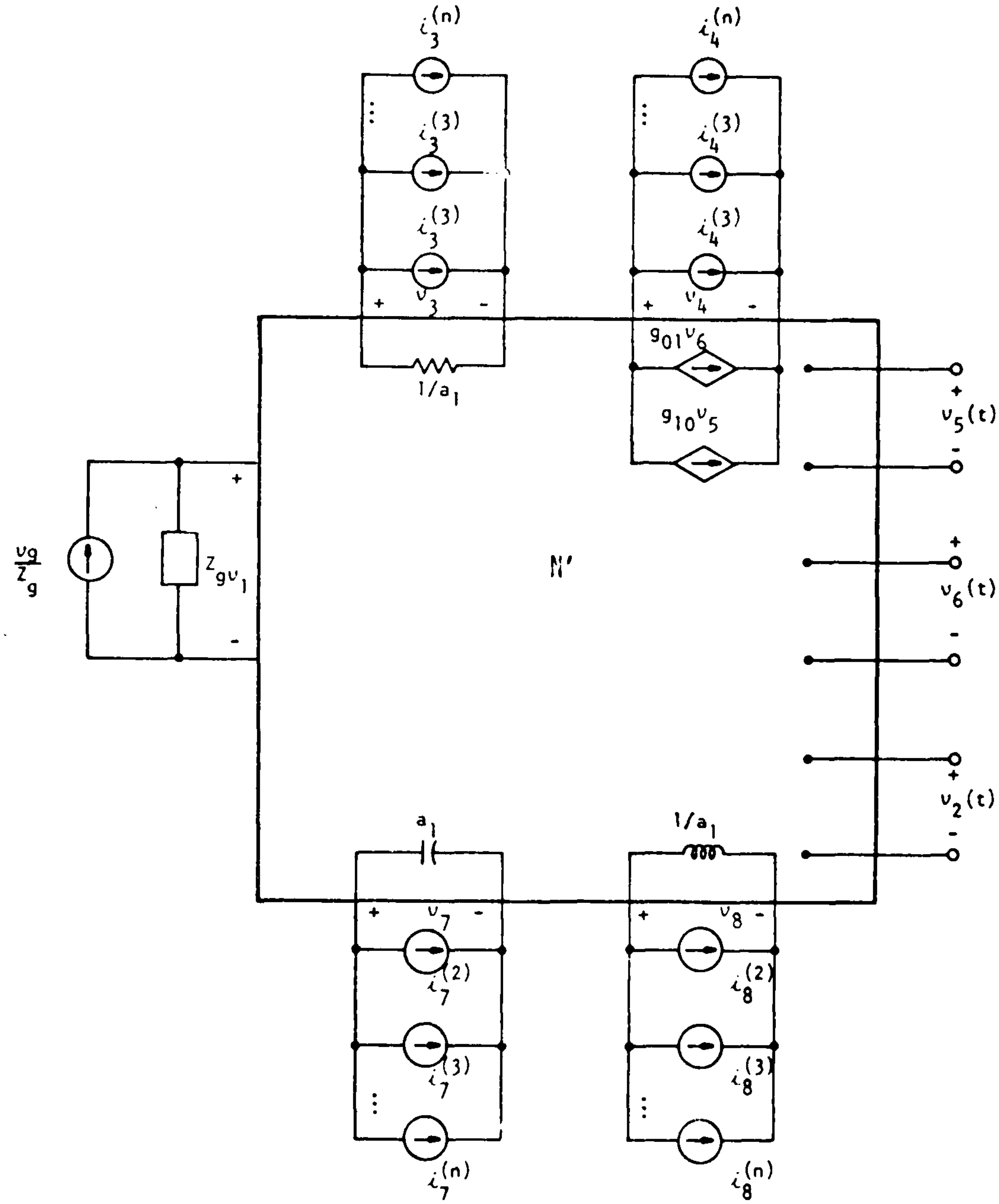


Figure 2-1. Steps in Nonlinear Circuit Analysis using Volterra Series.



(c)

Figure 2-1. (contd.)

The output variables to be found are the voltages at these eight ports. The augmented linear network is denoted by N' in Fig. 2-1(c). To solve for the voltage vector $\underline{v} = [v_1\ v_2\ v_3\ \ldots\ v_8]$, we immediately recognize that the branches across these voltage variables must be selected as part of the tree [20]. Clearly, some of the other branches in the augmented linear network may also appear as part of the tree. These will then appear as voltage variables in the cutset equations for the augmented linear network. Since there is no need for these additional variables, we can reduce the dimensionality of our equations by a systematic elimination of these unwanted variables. In the case under consideration, we should be left with only the vector $\underline{v} = [v_1\ v_2\ \ldots\ v_8]$ as the unknown vector. Each of these 8 ports will have a set of transfer functions of order 1 to n associated with it. Our task here is to solve for these transfer functions.

At this point, we make the following general notational definitions:

$$\underline{H}_k(s_1,s_2,\ldots,s_k) = \begin{bmatrix} H_k^1(s_1,\ldots,s_k) \\ H_k^2(s_1,\ldots,s_k) \\ \cdot \\ \cdot \\ \cdot \\ H_k^m(s_1,s_2,\ldots,s_k) \end{bmatrix} \tag{2-50}$$

where

$H_k^j \equiv$ kth order nonlinear transfer function from the input to the jth port; $m = 8$ in our example here.

$$\underline{v}(t) = [v_1(t)\ v_2(t)\ \dots\ v_m(t)]^T \tag{2-51}$$

where $v_i \equiv$ voltage at the ith port

The cutset equations for the m-port nonlinear network can be written as:

$$\underset{\sim}{Y}(p)\underline{v} + \underline{F(\underline{v})} + \underline{G(u,v)} + p\underline{Q}(\underline{v}) + \frac{1}{p}\underline{\phi}(\underline{v}) =$$

$$[v_g/z_g(p)][1\ \ 0\ \ 0\ \ \dots\ \ 0]^T \tag{2-52}$$

where

$p \equiv$ differential operator, $\frac{d}{dt}$

$\underset{\sim}{Y}(p) \equiv$ Reduced admittance matrix for the p-port augmented linear network

$\underline{F}(\underline{v}) \equiv$ vector composed of all nonlinear currents through the zero memory independent nonlinearity

$\underline{G(u,v)} \equiv$ vector composed of all nonlinear currents through the zero memory dependent nonlinearities

$\underline{Q}(\underline{v}) \equiv$ vector composed of all nonlinear currents through the nonlinear capacitive nonlinearities

$\underline{\phi}(\underline{v}) \equiv$ vector composed of all nonlinear currents through the nonlinear inductive elements.

$z_g(p) \equiv$ source impedance

Since the linear parts of the functions F(•), G(•), Q(•), and ϕ(•) in eqn. (2-46) through (2-49) have been lumped together with the linear part of the network, the general form of these functions will be as follows:

$$\underline{Z}(\underline{v}) = \underline{Z}_2(\underline{v}) + \underline{Z}_3(\underline{v}) + \underline{Z}_4(\underline{v}) + \cdots \tag{2-53}$$

where

$Z_2(\underline{v})$ is a quadratic function of $\underline{v}$

$Z_3(\underline{v})$ is a cubic function of $\underline{v}$

$Z_4(\underline{v})$ is a quartic function of $\underline{v}$

$\cdots$

$\underline{Z}(\cdot)$ being $\underline{F}(\cdot)$, $\underline{G}(\cdot)$, $\underline{Q}(\cdot)$, or $\phi(\cdot)$. Thus, eqn. (2-53) can be re-written as:

$$\underset{\sim}{Y}(p)\underline{v} = \frac{1}{z_g(p)}\begin{bmatrix} v_g(t) \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} - \underline{i}_k(t) \qquad , k \geq 2 \tag{2-54}$$

where $\underline{i}_k(t)$ denotes vectors of 2nd and higher order current sources due to $\underline{F}(\underline{v})$, $\underline{G}(\underline{u},\underline{v})$, $pQ(\underline{v})$, and $\frac{1}{p}\phi(\underline{v})$. The mathematical artifice used in section (2-2) could have been applied here also to obtain the form of all the nonlinear current source terms, $\underline{i}_k(t)$. For the sake of brevity, we will not use that approach here, but simply use the results of section (2-2) to identify the different order current sources due to different nonlinearities. These are summarized in Table 2-1, where $v^i(t)$ denotes the ith order response voltage v(t), which control the nonlinear element characteristics.

Table 2-1. Nonlinear Current Sources in *multiple-node*, multiple-nonlinearity circuit analysis.

Nonlinear Resistor, F(v):

k = 2: $a_2[v^1]^2$

k = 3: $2a_2[v^1v^2]+a_3[v^1]^3$

k = 4: $a_2[2v^1v^3+(v^2)^2]+3a_3[v^1]^2v^2+a_4[v^1]^4$

Nonlinear Dependent Nonlinearity G(u,v):

k = 2: $a_{20}[u^1]^2 + a_{02}[v^1]^2 + a_{11}u^1v^1$

k = 3: $a_{30}[u^1]^3 + a_{03}[v^1]^3 + a_{21}[u^1]^2v^1 + a_{12}u^1[v^1]^2 + 2a_{20}u^1u^2 + 2a_{02}v^1v^2 + a_{11}[u^1v^2 + u^2v^1]$

k = 4: $a_{40}[u^1]^4 + a_{04}[v^1]^4 + a_{13}u^1[v^1]^3 + a_{22}[u^1]^2[v^1]^2 + a_{21}(2u^1u^3 + [u^2]^2) + a_{11}(u^3v^1 + u^1v^3 + u^2v^2) + a_{02}(2v^1v^3 + [v^2]^2) + 3a_{30}[u^1]^2v^2 + 3a_{03}[v^1]^2v^2 + a_{21}([u^1]^2v^2 + 2u^1u^2v^1) + a_{12}(u^2[v^1]^2 + 2u^1v^1v^2)$

Nonlinear Capacitive Nonlinearity pQ(v):

k = 2: $a_2p[v^1]^2$

k = 3: $2a_2p[v^1v^2]+a_3p[v^1]^3$

k = 4: $a_2p(2v^1v^3+[v^2]^2)+3a_3p[v^1]^2v^2+a_4p[v^1]^4$

Table 2-1 (contd.)

Nonlinear Inductive Nonlinearity, $[1/p]\phi(v)$

$k = 2$: $\frac{a_2}{p}[v^1]^2$

$k = 3$: $\frac{2a_2}{p}[v^1v^2]+\frac{a_3}{p}[v^1]^3$

$k = 4$: $\frac{a_2}{p}(2v^1v^3+[v^2]^2)+\frac{3a_3}{p}[v^1]^2v^2+\frac{a_4}{p}[v^1]^4$

We observe that the nonlinear current source terms in Table 2-1 are similar to the nonlinear terms whose transforms were derived in section 2-2, except for the nonlinear dependent source terms, which are functions of two controlling voltages u and v. The form of the transforms of the nonlinear dependent source will, however, be similar to the other nonlinearity types. These can again be written by inspection. For example,

$$a_{20}[u^1(t)]^2 \rightarrow a_{20}u^1(t_1)u^1(t_2) \leftrightarrow a_{20}\ U(s_1)U(s_2) \tag{2-55}$$

$$a_{11}u^1(t)v^1(t) \rightarrow a_{11}u^1(t_1)v^1(t_2) \leftrightarrow a_{11}\ \overline{U(s_1)V(s_2)} \tag{2-56}$$

$$a_{20}u^1(t)v^2(t) \rightarrow a_{20}u^1(t_1)v^2(t_2,t_3) \leftrightarrow a_{20}\ \overline{U(s_1)V(s_2,s_3)} \tag{2-57}$$

Recall the one way arrow goes backwards only when $t_1=t_2=t_3$.

We also note that a k-th order current source term in Table 2-1 depends on responses of order less than k, which implies that, in order to calculate a transfer function of order k, we need to determine the transfer function up to order (k-1).

The first order transfer function can be solved for easily. It is simply the linear circuit response. Therefore,

$$\underset{\sim}{Y}(p)\underline{v}(t) = \underline{i}_1(t) \tag{2-58}$$

For a single input system, $\underline{i}_1(t) = 1/z_g\ [v_g(t)\ \ 0\ \ 0\ \ \ldots\ \ 0]^T$, where $v_g(t)$ is the source voltage. Taking the transform of eqn. (2-58), and assuming that the input source to be an impulse function, we get:

$$\underline{V}^1(s_1) = \underline{H}_1(s_1) = 1/z_g\ [\underset{\sim}{Y}(s_1)]^{-1}[1\ \ 0\ \ 0\ \ \ldots\ \ 0]^T \tag{2-59}$$

where $\underline{H}_1(s_1)$ was defined in eqn. (2-50).

The equation for obtaining the second-order response, as per eqn. (2-54), is the following:

$$\underset{\sim}{Y}(p)\underline{v}^{(2)}(t) = -\underline{i}_2(t) \qquad (2\text{-}60)$$

Since the input to the nonlinear circuit is assumed to be an impulse function, the transform of eqn. (2-60), after using eqn. (2-14), is:

$$\underset{\sim}{Y}(s_1+s_2)\underline{H}_2(s_1,s_2) = -\underline{I}_2(s_1,s_2) \qquad (2\text{-}61)$$

The elements of vector $\underline{I}_2(s_1,s_2)$ can be obtained by performing a two-dimensional transform on the terms associated with k = 2 in Table 2-1. This operation, as indicated earlier, can be carried out by inspection. Thus, we have

$$\underline{H}_2(s_1,s_2) = -[\underset{\sim}{Y}(s_1+s_2)]^{-1}\underline{I}_2(s_1,s_2) \qquad (2\text{-}62)$$

Likewise we can solve for $\underline{H}_3(s_1,s_2,s_3)$. In general, we solve for the nth order transfer function using eqn. (2-63):

$$\underline{H}_n(s_1,s_2,\ldots,s_n) = [\underline{Y}(\sum_{i=1}^{n} s_i)]^{-1}\underline{I}_n(s_1,\ldots,s_n) \qquad (2\text{-}63)$$

We observe a striking similarity between eqn. (2-63) and the equations for nodal or cutset analysis encountered in linear circuit analysis. A little thought would show that the process of solving eqn. (2-63) is identical to solving the linear circuit in Fig. 2-2. We have nonlinear current sources as inputs to the augmented linear circuit. A k-th order vector of transfer functions is obtained by exciting the linear circuit by the kth order current sources. Just as in the case of linear systems, superposition can be applied here when a particular order response is determined from the

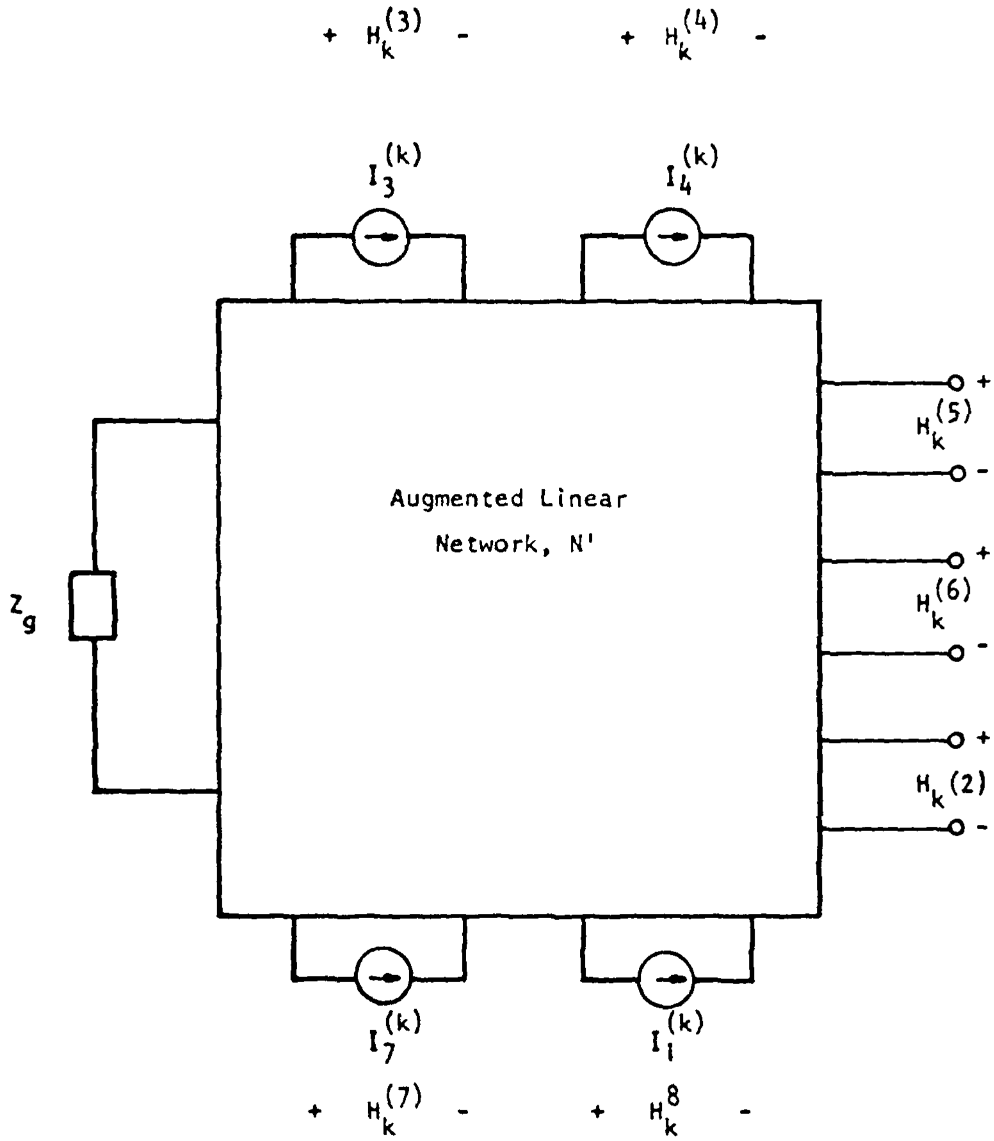


Figure 2-2. Determination of Volterra Transfer Functions.

lower order responses. That is, a k-th order response can be obtained by applying the k-th order current sources one-by-one at each of the ports ar then summing up the responses. It is important to note, however, that the complete responses of order up to (k-1) must be determined before we can obtain the kth order response by superposition. It is also noted that the illustration of Fig. 2-2 is for pedagogic purpose and that the nonlinear current sources are not physically present in the circuit under consideration.

2-5. Multiple Input Circuit Analysis

Much of the foregoing discussion has been concerned with the analysis of nonlinear circuits with single inputs. However, many applications of practical significance is nonlinear circuit analysis have multiple inputs. For example, in a receiver system, the mixer circuit has two inputs: 1) the message signal, and 2) the local oscillator signal. The transmitter again has nonlinear circuits with multiple inputs. The Volterra series method is especially well suited for the analysis of such circuits. In this section we discuss how the various order transfer functions change as a result of multiple inputs.

From the discussion in section 2-4, it should be apparent that the analysis of nonlinear circuits using the Volterra series method involves the repeated analysis of a linearized circuit. The fundamental relationship had the following form (see eqn. 2-54):

$$\underset{\sim}{Y}(p)\underline{v}(t) = \frac{1}{z_g(p)}\,\underline{i}_1(t) - \underline{i}_2(t) - \underline{i}_3(t) + \ldots \qquad (2\text{-}64)$$

where $\underline{i}_k(t)$ is the k-th current source vector. For $k \geq 2$ the k-th order current source, depends on up to the (k-1) order voltage ratio transfer functions as discussed above. It is injected at each of the pots at which the nonlinear elements are present, and is due entirely to the nonlinear characteristics of the nonlinearity. Furthermore, it is proportional to the k values of the circuit input multiplied together. Thus, the number of elements in the vector $\underline{i}_k(t)$, $k \geq 2$, remain unchanged when multiple inputs are present; only the $\underline{i}_1(t)$ vector is changed.

Consider, for example, the two-input circuit of Fig. 2-3(a). Then, to solve for the first-order transfer function, we write the vector transform equation as:

$$\underset{\sim}{Y}(s_1)\underline{V}(s_1) = \underline{I}_1(s_1) \tag{2-65}$$

where

$$I_1(s_1) = [Y_{g1}(s_1)V_{g1}(s_1) \quad Y_{g2}(s_1)V_{g2}(s_1) \quad 0 \; \ldots \; 0]^T \tag{2-66}$$

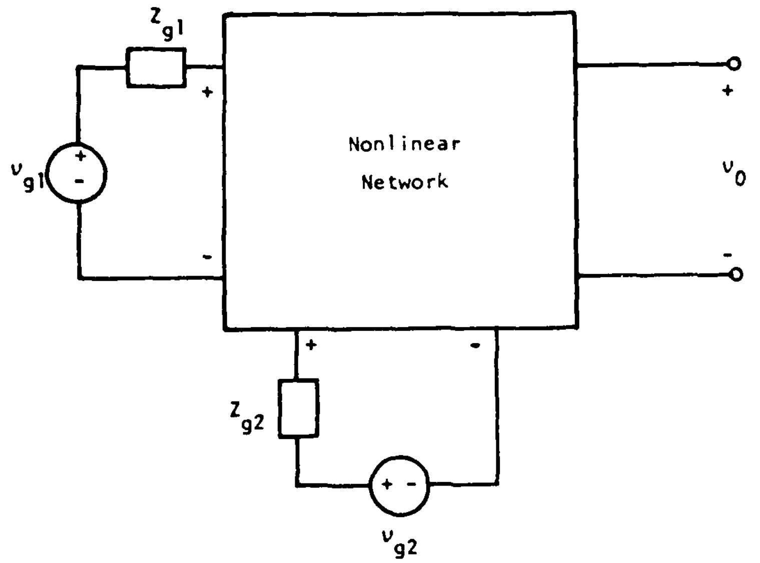
and $\underset{\sim}{Y}$ and $\underline{V}$ are as defined previously. The transfer function vector can be written as:

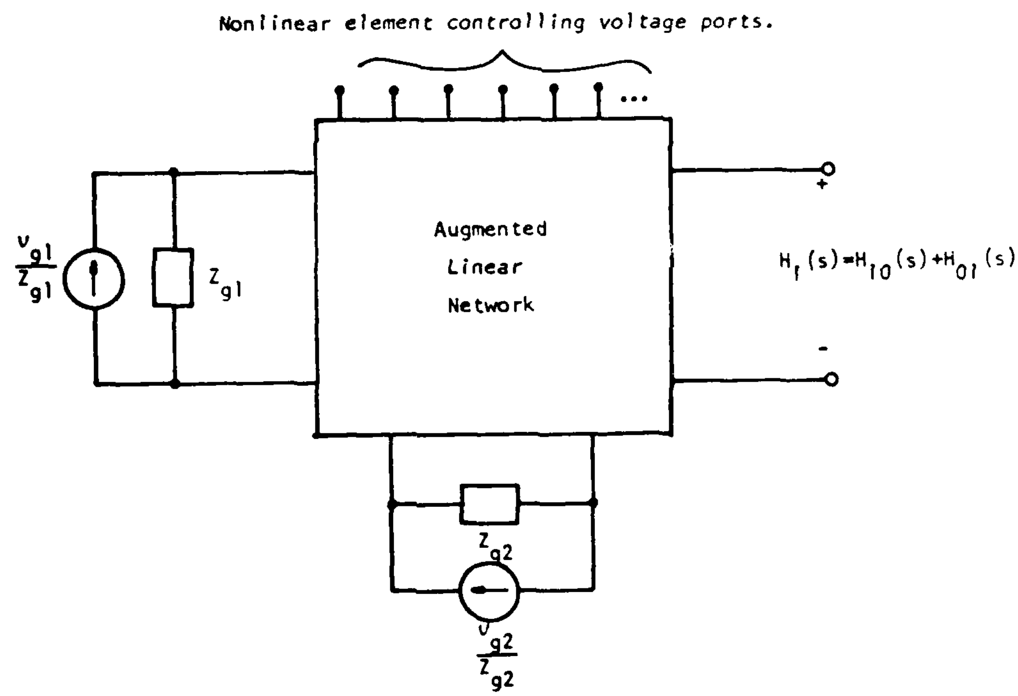
$$\underline{H}_1(s_1) = \underline{H}_{10}(s_1) + H_{01}(s_1) \tag{2-67}$$

where

$$\underline{H}_{10}(s_1) = \left[\frac{V^{(1)}(s_1)}{V_{g1}} \; \frac{V^{(2)}(s_1)}{V_{g1}} \; \cdots \; \frac{V^{p}(s_1)}{V_{g1}}\right]^T \Bigg|_{V_{g2}=0} \tag{2-68}$$

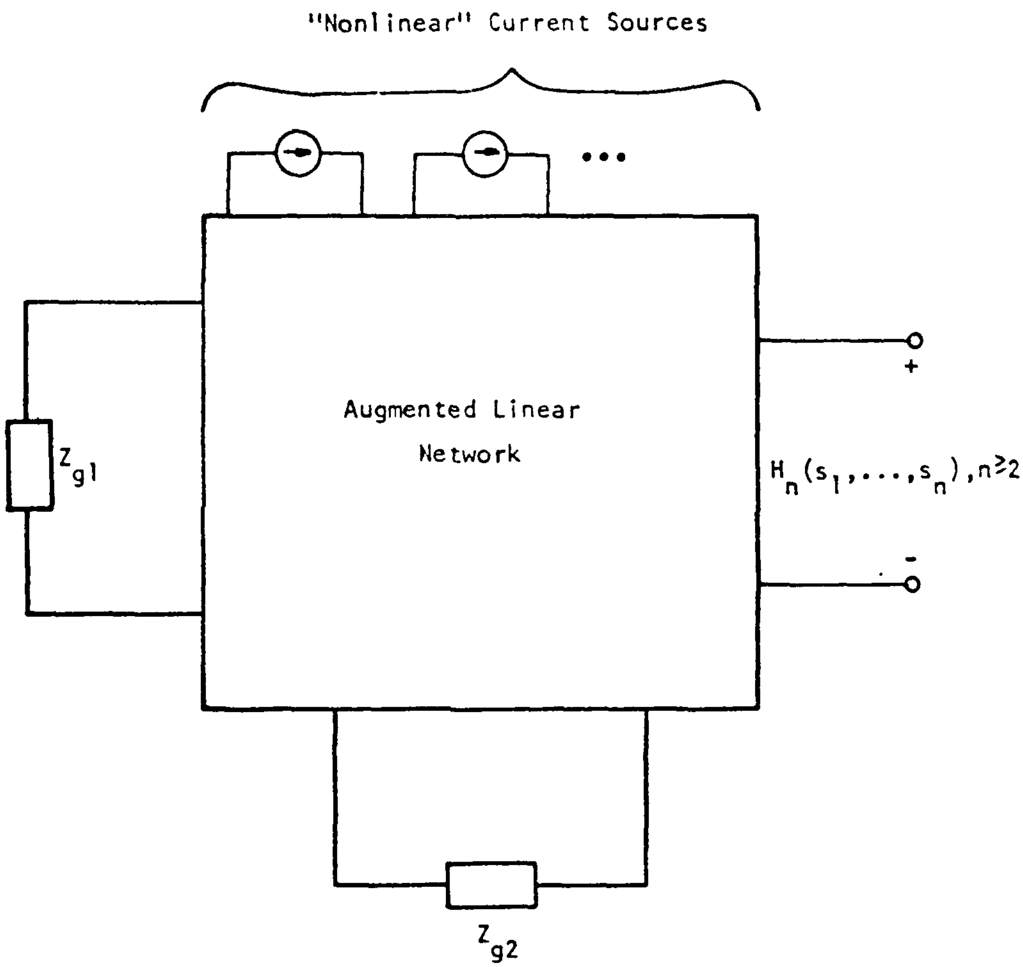
and





(b) Circuit for determining first-order transfer function.

Fig. 2-3. Multiple Input Nonlinear Circuit Analysis.



(c) Circuit for determining
$H_n(s_1,s_2,...,s_n), n\geq 2$.

Fig. 2-3. (contd.) Multiple Input Nonlinear Circuit Analysis.

$$\underline{H}_{01}(s_1) = \left[\frac{V^{(1)}(s_1)}{V_{g2}} \;\; \frac{V^{(2)}(s_1)}{V_{g2}} \cdots \frac{V^{(p)}(s_1)}{V_{g1}}\right]^T \Bigg|_{V_{g1}=0} \qquad (2\text{-}69)$$

where $V^{(i)}$ is the voltage at port i.

The second- and higher-order transfer function vectors are solved for by removing the given input sources and applying the fictitious nonlinear current sources across the ports at which the nonlinear elements are present. The vector transform equation for solving for the second-order transfer function is still given by:

$$\underline{H}_2(s_1,s_2) = -[\underset{\sim}{Y}(s_1 + s_2)]^{-1} [\underline{I}_2(s_1,s_2)] \qquad (2\text{-}70)$$

where

$$\underline{I}_2(s_1,s_2) = [I^{(1)}(s_1,s_2) \; I^{(2)}(s_1,s_2) \; \ldots \; I^{(p)}(s_1,s_2)] \qquad (2\text{-}71)$$

Depending on the nonlinearity type, the general form of $I^{(\ell)}(s_1,s_2)$, the second-order current source across port ℓ, will be:

$$I_2^{(\ell)}(s_1,s_2) = a_2 H_1^{(\ell)}(s_1) H_1^{(\ell)}(s_2) \qquad (2\text{-}72)$$

where $H_1^{(\ell)}(\cdot)$ is known from eqn. (2-67). The determination of the higher-order transfer functions is done similarly.

In summary, we note that the presence of multiple input sources in a nonlinear circuit does not drastically alter the procedure for determining the Volterra transfer functions. Only the structure of the first-order current source vector is changed as a result of multiple sources. This change is reflected in the values of the elements making up the second- and higher-order current source vectors, whose structure remains unchanged.

2-6. Sinusoidal Steady-State Analysis

In linear system theory, the sinusoidal steady-state response is intimately tied to the transfer function of the system. A similar result is found for higher order responses using the Volterra series method: an n-th order response at a particular frequency is directly related to the n-th order transfer function. In this section we develop this relationship.

If the harmonic input method [10-12] had been used in deriving the generalized transfer functions in the previous sections, the relationship between the n-th order steady state response and the n-th order transfer function would have been self-evident. But, since multi-dimensional transform theory was used to derive the generalized transfer functions, this relationship must be developed. We treat the specific case of n=2 in section 2-6.1 and then derive the general relationship in section 2-6.2.

2-6.1. Second-order Sinusoidal response:

The second-order output, according to the Volterra series, is given by:

$$y_2(t) = \int_0^\infty \int_0^\infty h_2(t-\tau_1, t-\tau_2)x(\tau_1)x(\tau_2)d\tau_1 d\tau_2 \qquad (2\text{-}73)$$

Consider the input signal comprising two unit sinusoidal signals at frequencies ω_a and ω_b. The input $x(\tau)$ is therefore:

$$x(\tau) = \left[\frac{\exp(j\omega_a\tau)+\exp(-j\omega_a\tau)}{2}\right] + \left[\frac{\exp(j\omega_b\tau)+\exp(-j\omega_b\tau)}{2}\right] \qquad (2\text{-}74)$$

Substituting eqn. (2-74) in (2-73), we have:

$$y_2(t) = \int_0^\infty\int_0^\infty h_2(t-\tau_1, t-\tau_2) \cdot$$

$$\cdot \left[\frac{\exp(j\omega_a\tau_1)+\exp(-j\omega_a\tau_1)}{2} + \frac{\exp(j\omega_b\tau_1)+\exp(-j\omega_b\tau_1)}{2}\right]$$

$$\cdot \left[\frac{\exp(j\omega_a\tau_2)+\exp(-j\omega_a\tau_2)}{2} + \frac{\exp(j\omega_b\tau_2)+\exp(-j\omega_b\tau_2)}{2}\right]$$

$$\cdot\, d\tau_1 d\tau_2 \qquad (2\text{-}75)$$

Considering one cross term only,

$$\int_0^\infty\int_0^\infty h_2(t-\tau_1, t-\tau_2)\, \frac{1}{4} \exp(j\omega_a\tau_1 + j\omega_b\tau_2) d\tau_1 d\tau_2 \qquad (2\text{-}76)$$

and letting $\sigma_1 = t-\tau_1$ and $\sigma_2 = t-\tau_2$ and carrying out the integration yields,

$$\frac{1}{4} H_2(j\omega_a, j\omega_b)\exp[j(\omega_a+\omega_b)t] \qquad (2\text{-}77)$$

Considering the other cross term similarly yields

$$\frac{1}{4} H_2(j\omega_b, j\omega_a)\exp[j(\omega_a+\omega_b)t] \qquad (2\text{-}78)$$

However, if $H_2(s_1, s_2)$ is symmetrical in its arguments, as they are assumed to be in this report, then the terms in eqns. (2-77) and (2-78) are equal. The complex conjugate terms appear similarly. Hence, the output at frequency $\omega_a+\omega_b$ is:

$$y(t)\big|_{\omega_a+\omega_b} = |H_2(j\omega_a, j\omega_b)|\cos[(\omega_a+\omega_b)t + \theta_{a+b}] \qquad (2\text{-}79)$$

The $2\omega_a$ or $2\omega_b$ term and their complex conjugates appear only once in eqn. (2-75); hence, their magnitude will be $\frac{1}{2}|H_2(j\omega_a,j\omega_a)|$ and $\frac{1}{2}|H_2(j\omega_b,j\omega_b)|$, respectively. If only one frequency input was present, the results would be similar. The second-order output would then be:

$$y_2(t) = |H_2(j\omega_a,-j\omega_a)| + \frac{|H_2(j\omega_a,j\omega_a)|}{2} \cos(2\omega_a t+\theta_{2a}) \qquad (2\text{-}80)$$

Thus, if we know $H_2(s_1,s_2)$, then the quantities in eqn. (2-80) can be easily evaluated. This is analogous to the case of linear systems, where the complex variable s is replaced by $j\omega$ to compute the response at ω.

If more than two-tones were present at the input, the second order response would be evaluated by taking all combinations of two frequencies at a time.

The response of the third and higher orders is similarly treated. We now present the general case.

2-6.2. General Sinusoidal Steady-State Analysis.

In this sub-section, we develop the relationship which can be applied directly to compute the sinusoidal steady-state response of a nonlinear system from its nonlinear transfer functions, which can be obtained by the method presented in section 2. The discussion here relies heavily on [10].

Consider a nonlinear system excited by the sum of K distinct tones; i.e., defining N = 2K, we have,

$$x(t) = \frac{1}{2} \sum_{i=1}^{N} A_i \exp(j\omega_i t) \qquad (2\text{-}81)$$

where ω_i will include both positive and negative frequencies, and A_i for a negative frequency will be the complex conjugate of A_i for the positive frequency in order to have x(t) real. Then, the nth order output, $y_n(t)$, is given by:

$$y_n(t) = \underset{n\text{-fold}}{\int \cdots \int} h_n(\tau_1,\ldots,\tau_n) \prod_{i=1}^{n} x(t-\tau_i)d\tau_i$$

$$= \int \cdots \int h_n(\tau_1,\ldots,\tau_n) \frac{1}{2^n} \sum_{i=1}^{n} \sum_{k=1}^{N} A_k \exp[j\omega_k(t-\tau_i)]d\tau_i \tag{2-82}$$

Carrying out the product operation in eqn. (2-82), we get a function $y_n(t)$ containing N^n terms, given by:

$$y_n(t) = \sum_{k_1=1}^{N} \cdots \sum_{k_n=1}^{N} \frac{1}{2^n} A_{k_1} \cdots A_{k_n} H_n(j\omega_{k_1},\ldots,j\omega_{k_n})$$

$$\cdot \exp[j(\omega_{k_1}+\ldots+\omega_{k_n})t] \tag{2-83}$$

Notice that in arriving at eqn. (2-83), we have performed the τ_i integration in eqn. (2-82), thus giving rise to the n-th order transfer function in eqn. (2-83). As the indices k_i are varied over the range 1 to N, many of the terms will be at the same frequency. The number of terms at various particular frequencies will vary according to what frequency combinations are taken. For example, in the case of n=2 in section 2-6.1, there were two cross frequency terms, while there was only one second harmonic (at $2\omega_a$) term. Similarly, for n=3, there are six terms in eqn. (2-83) at frequency $\omega_a+\omega_b+\omega_c$, three terms at $2\omega_a+\omega_b$, one term at $3\omega_a$, etc. The nonlinear transfer functions, which make up the coefficients of these frequency terms, differ only in their arguments. However, since the transfer functions are assumed to be symmetric, the coefficient of the output at frequency $\omega_a+\omega_b+\omega_c$

(in the case of n=3) can be multiplied by 6. This obviates the need for taking all combinations to compute the output at $\omega_a+\omega_b+\omega_c$. Likewise we handle the case of other frequency combinations. With this insight, we can peek at the problem from a different perspective.

Let $m_1, m_2, \ldots, m_N$ be non-negative integers. Then, the number of terms at frequency $\omega_\Sigma = m_1\omega_1+m_2\omega_2+\ldots+m_N\omega_N$ is equal to the number of ways of forming $m_1\omega_1+\ldots+m_N\omega_N$. In the n-th order output spectrum to a multi-tone input, each term is evaluated by taking a distinct combination of n input tones at a time. To compute the n-th order output when the input frequencies are $\omega_1, \omega_2, \ldots, \omega_N$, we must therefore restrict m_i in the following manner to compute ω_Σ:

$$m_1 + m_2 + \cdots + m_N = n \tag{2-84}$$

Now the problem reduces to the following: find the number of ways in which n objects can be divided into N groups of which the first contains m_1 objects, the second m_2 objects, etc. The solution to this problem is given by the multi-nomial coefficient [22]:

$$C_{n,N} = \frac{n!}{m_1!m_2\ldots m_N!} \tag{2-85}$$

By deriving eqn. (2-85), we have obviated the repetition of terms that is inherent in eqn. (2-83). An equivalent way of representing eqn. (2-83) through the use of eqn. (2-85) then becomes:

$$y_n(t) = \sum_{n,N} C_{n,N} \frac{A_1^{m_1} A_2^{m_2} \ldots A_N^{m_N}}{2^n}$$

$$\cdot\, H_n(\underbrace{j\omega_1,\ldots}_{m_1\text{times}},\underbrace{j\omega_2,\ldots}_{m_2\text{times}}\ \ldots,\ \underbrace{j\omega_N,\ldots}_{m_N\times})$$

$$\cdot\, \exp[j(m_1\omega_1 + \ldots + m_N\omega_N)t] \tag{2-86}$$

Since $y_n(t)$ is real, eqn. (2-86) also contains the complex conjugate terms. Thus, the coefficient of the sinusoidal term at frequency $m_1\omega_1+\ldots+m_N\omega_N$ in the n-th order output is given by:

$$C_{n,N}\ \frac{A_1^{m_1}A_2^{m_2}\ldots A_N^{m_N}}{2^{n-1}}\ H_n(\underbrace{j\omega_1,\ldots}_{m_1\text{times}},\underbrace{j\omega_2,\ldots}_{m_2\text{times}},\ldots,\underbrace{j\omega_N,\ldots,}_{m_N\text{times}}) \tag{2-87}$$

In computing the entire n-th order response in eqn. (2-86), we take all distinguishable combinations of m_i satisfying eq. 82-84). According to [10] there are

$$S_{n,N} = \binom{n+N-1}{n} = \frac{(n+N-1)!}{n!(N-1)!} \tag{2-88}$$

such combinations.

Equation (2-86) is the fundamental relationship between the n-th order output and the n-th order transfer function. At first glance, the evaluation of this equation appears to be a formidable task. But, after some thought, one finds that this is not such a difficult task after all. We, however, defer the discussion of this till section 4.

We now illustrate the use of eqn. (2-87). We assume that the nonlinear transfer functions are known. The case for n=2 can be easily verified from the discussion in section 2-6.1. For a two-tone input at ω_1 and ω_2 and n=3, we have the following cases:

(a) The output at ω_1 and ω_2 have the following amplitudes, respectively:

$$y_3(t)|_{\omega_1} = \frac{3!|A_2|^2 A_1}{(4)1!1!1!} |H_3(j\omega_1,-j\omega_2,j\omega_2)| \tag{2-89}$$

$$y_3(t)|_{\omega_2} = \frac{3!A_2|A_1|^2}{(4)1!1!1!} |H_3(j\omega_1,-j\omega_1,j\omega_2)| \tag{2-90}$$

(b) The output at $2\omega_1+\omega_2$ has the following magnitude:

$$y_3(t)|_{2\omega_1+\omega_2} = \frac{3!A_1^2 A_2}{(4)2!1!} |H_3(j\omega_1,j\omega_1,j\omega_2)| \tag{2-91}$$

(c) The output at $3\omega_1$ has the following magnitude:

$$y_3(t)|_{3\omega_1} = \frac{3!(A_1)^3}{(4)3!} |H_3(j\omega_1,j\omega_1,j\omega_1)| \tag{2-92}$$

The other combinations can be carried out similarly. For the above cases we make the following observations: both eqns. (2-89) and (2-90) are similar to obtaining the output at $\omega_a+\omega_b+\omega_c$, and therefore we see a 3! (=6) multiplication factor*, which accounts for the six combinations at $\omega_a+\omega_b+\omega_c$ that were mentioned earlier; eqn. (2-91) is similar to obtaining the output at $2\omega_a+\omega_b$, and therefore has a multiplication factor of (3!/2!) = 3, which again is in accordance with our earlier discussion; eqn. (2-92) is like evaluating the output at $3\omega_a$, and hence has a multiplication factor of (3!/3!) = 1.

In section (2-5), we dealt with the analysis of multiple input nonlinear circuits. In obtaining the sinusoidal steady-state response of such

*The constant factor 4 in the denominator appears consistently in all the output terms, and is therefore not regarded as a variable multiplication factor here. This factor appears due to the way x(t) was expressed in eqn. (2-81).

circuits the material of this section is still applicable. However, care must be taken in keeping track of the various input frequencies, and their associated transfer functions, when such an analysis is warranted.

CHAPTER 3

COMPUTER-AIDED ANALYSIS USING VOLTERRA SERIES

3-1. Introduction

The adapting of Volterra series method in a general simulation program has been regarded as difficult by various authors [30]. As such, virtually no effort has been spent on investigating the computational aspect of this method. Most previous works, such as [7], have endeavored to check the validity of this approach by applying it to specific circuit problems using a computer.

The only major effort in using the Volterra series for general nonlinear circuit analysis has been the development of the program NCAP [10,24]. A cursory review of this program reveals the inherent inefficiency in the computational approach with regards to storage and types of algorithms used. This inefficiency notwithstanding, there are severe limitation regarding the usefulness of the program: first, the program merely computes the numerical values of the nonlinear transfer function at the various program-prescribed combinations of the input frequencies, and does not compute all the transfer function values which are required to compute the complete output spectrum. Thus, NCAP does not yield the entire output spectrum information. Second, to compute up to an n-th order transfer function, the user must specify n input frequencies, which are assumed to be a sum of exponentials and not real sinusoids. The program, therefore, is severely limited in its usefulness from the point of view of a user who may only be interested in obtaining the output spectrum - say, for example, up to the third order response to two sinusoidal inputs - and has little use for the numerical values of the transfer functions at the program prescribed frequencies.

In this section we look at the computational aspect of the Volterra series method for general simulation purposes and then present the basic algorithms for adapting this method for the spectrum and distortion analysis of nonlinear circuits with polynomial type nonlinearities.

In section 3-2, we present a brief overview of symbolic analysis in linear circuits, and then describe the reason why a symbolic approach is particularly useful in adapting Volterra series for general simulation. Section 3-3 deals with the implementation of the symbolic approach, and also contrasts the computational effort between a numerical approach and the particular symbolic approach used here. The algorithm for obtaining the complete output spectrum and the various distortion indices is described in section 3-4. A description of the computer implementation of these algorithms is given in section 3-5.

3-2. Why a Symbolic Analysis Approach.

The symbolic analysis of circuits involves the computation of the a_i and b_i for network functions in the form

$$F(s) = \frac{N(s)}{D(s)} = \frac{\sum a_i s^i}{\sum b_i s^i} \qquad (3\text{-}1)$$

when all circuit elements are known. The more general form

$$F(s;x_1,x_2,\ldots,x_n) = \frac{N(s;x_1,\ldots,x_n)}{D(s;x_1,\ldots,x_n)} \qquad (3\text{-}2)$$

applies when some elements of the circuit x_i are kept as symbols. The advantages of symbolic analysis have been recognized previously [25,27]. One particular advantage, and the one which is relevant to our problem here, is that the numerical evaluation of a function at discrete points is much easier and faster once the symbolic function is obtained than working repeatedly with a circuit analysis program. With this brief overview of symbolic analysis, we now proceed to answer the question: Why use a symbolic analysis approach for adapting the Volterra series method for general circuit analysis?

As pointed out in the previous sections, a nonlinear circuit is completely characterized by its Volterra kernels, or their transforms - the generalized transfer functions. These transfer functions are then directly related to the various order sinusoidal steady-state responses, as described in Chapter 2. The n-th order transfer function is determined from the following equation (see Chapter 2):

$$\underline{H}_n(s_1,\ldots,s_n) = [\underset{\sim}{Y}(\sum_{i=1}^{n} s_i)]^{-1}\ \underline{I}_n(s_1,\ldots,s_n) \qquad (3\text{-}3)$$

where $\underset{\sim}{Y}(\sum_{i=1}^{n} s_i)$ is the reduced node admittance matrix evaluated as $s_1 + s_2 + \ldots + s_n$, and $\underline{I}_n$ is the n-th order current source vector due to the nonlinear elements. To compute the output spectrum, we evaluate H_n at the various and many frequency combinations. From eqn. (3-3) it should be clear that such an evaluation will entail the inversion of the reduced node admittance matrix at each of these frequency combinations. Using combinatorial analysis, it has been shown [22] that for an input consisting of M sine waves, the number of inversions involved in an n-th order response, given by $N_{n,m}$, is:

$$N_{n,m} = \binom{2M+n-1}{n} \qquad (3\text{-}4)$$

Thus, for a 3-tone input and up to a third order analysis, the number of inversions is approximately 285. For higher order responses, this number grows very rapidly.

Two basic approaches available for handling this inversion process are: 1. Numerical approach, or 2. Symbolic approach. The advantage of evaluating symbolic transfer functions mentioned earlier makes the symbolic approach more attractive. How much advantage is gained in using a symbolic analysis depends on how much computational effort is expended in obtaining the symbolic inverse of the reduced node admittance matrix. Thus, an efficient scheme for obtaining the symbolic inverse must be used to efficiently adapt the Volterra series method for computer aided analysis. The determination of the symbolic inverse will be the subject of section 3-3.

The reasons presented above stem from looking at the computational aspect of adapting Volterra series for computer-aided analysis. There are other advantages gained from using a symbolic analysis. An important one is that the generalized transfer functions can be obtained as functions of s_i once the inverse of the reduced node admittance matrix is obtained as a symbolic function of s. This can be seen from examining eqn. (3-3). The formation of the n-th order current source vector is a bootstrapping operation, as was pointed out in Chapter 2. That is, an n-th order source is formed from transfer functions of order less than n. The first-order transfer function vector is determined from a column* of the symbolic inverse of the reduced node admittance matrix. The second order current sources, which

*This is assuming a single input circuit.

depend on the elements of the first order transfer function vector, are therefore formed from this column of $[Y(s)]^{-1}$. The second-order transfer function vector is obtained by pre-multiplying the second-order current source vector by $[Y(s_1+s_2)]^{-1}$, according to which the second-order transfer function vector eventually depends on the entries of inverse of the node admittance matrix evaluated at (s_1+s_2). The third- and higher-order transfer functions have a similar dependence. Thus, an inverse of the reduced node admittance matrix in symbolic form, with s retained as a symbol, also yields a functional description of the nonlinear transfer functions. A concomitant advantage of this functional description is that theorems from multidimensional theory [5] (such as initial value, final value, etc.) can then be used to gain more insight into the workings of the circuit.

[23] has developed recursive relationships to estimate the error incurred in the truncation of the series solution. This error was directly related to the L_1 norm of the linear kernel function, which, in turn, is related to the poles and residues of the linearized system. Thus, we can get an estimate of the accuracy of our solution through the pole-residue information provided to us by the symbolic analysis.

3-3. Symbolic Analysis Method

Symbolic circuit analysis by digital computer has been of considerable interest in the past decade. Many algorithms and methods have been derived to obtain symbolic transfer functions of linear circuits [20]. Most of these methods use tree enumeration [26], signal-flow graphs [20], or purely numerical methods [27] to obtain symbolic transfer function between the input and the output. These approaches are basically useful for single-input, single-output systems. The inversion of the reduced node admittance matrix to obtain the open-circuit impedance matrix, which is the problem we are dealing with, is basically a multi-input, multi-output problem. The methods mentioned above can be adapted for solving the problem at hand; however, the generation of multiple symbolic functions using these approaches many not be satisfactory because of excessive computer time requirements. Some other approach is definitely warranted.

Published methods [16-18] for inverting the nodal admittance matrix when the elements are rational functions of the Laplace transform variable s use pivotal techniques. It may appear that, since it is easy to program a computer to perform polynomial arithmetic, these pivotal-techniques are a natural way to approach the symbolic inversion problem. Results from the use of such a technique have proved to be disappointing, mainly due to the following reasons:

(a) The process of inversion transforms the nodal admittance matrix, which contains terms of the form $as + \frac{b}{s} + c$, into a matrix in which every element is a rational function of s. The pivotal technique produces the inverse matrix where common factors appear between numerator and denominator, and unless some mechanism is built into the process whereby these common factors are recognized and removed, the elements produced will have polynomials of excessively high order.

(b) When the circuit complexity is high, the evaluation of the symbolic function at high frequency values can give rise to numerical problems. For example, a circuit with 8 poles will have an s^8 term in the characteristic polynomial. When evaluated at 10 Mrad/sec, this term produces a number equal to 10^{56}. Of course, this problem can be alleviated by obtaining a partial fraction expansion (PFE) form for the transfer functions. But this again entails additional computations - not to mention the numerical instability problems involved in root finding.

(c) It has also been found that pivotal techniques become numerically unstable for higher order circuits.

We therefore seek another alternative for obtaining the symbolic form of the open circuit impedance matrix.

An approach based on the state variable formulation can be used to achieve this goal. Specifically, consider the general p-port augmented linear circuit of Fig. 3-1(a). We wish to solve for the transfer impedances, $z_{ij}(s)$, $i,j = 1,2,\ldots,p$, from the j-th port to the i-th port. Knowing these transfer impedances, we can write for the p-port:

$$\underline{V}(s) = \underset{\sim}{Z}(s)\underline{I}(s) = [\underset{\sim}{Y}(s)]^{-1}\,\underline{I}(s) \tag{3-5}$$

where $$\underline{V}(s) = [V_1(s)\ V_2(s)\ \ldots\ V_p(s)] \tag{3-6}$$

$$\underset{\sim}{Z}(s) = [z_{ij}(s)] \tag{3-7}$$

and $$\underline{I}(s) = [I_1(s)\ I_2(s)\ \ldots\ I_p(s)] \tag{3-8}$$

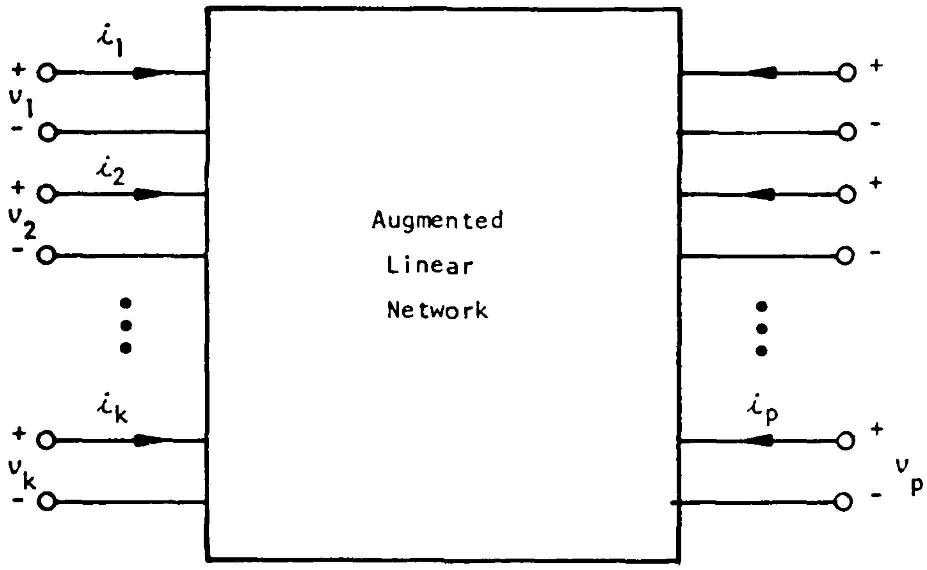
Note that the vector $\underline{V}(s)$ contains entries which are the output voltages and voltages that control the nonlinear element characteristics in the nonlinear circuit.

To obtain $\underset{\sim}{Z}(s)$ symbolically, we write for the network of Fig. 3-1(b), the following state equations:

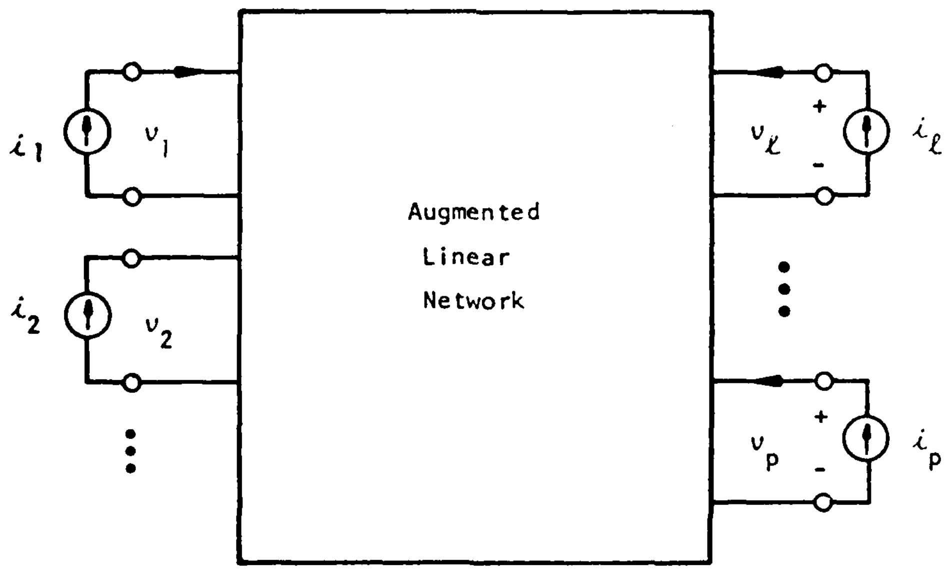
$$\dot{\underline{x}} = \underset{\sim}{A}\underline{x} + \underset{\sim}{B}\underline{i} \tag{3-9}$$

$$\underline{v} = \underset{\sim}{C}\underline{x} + \underset{\sim}{D}\underline{i} \tag{3-10}$$

where $\underline{x}$ is the vector of state variables, and $\underline{v}$ and $\underline{i}$ are vectors whose



(a)



(b)

Figure 3-1. Determination of $[Y(s)]^{-1}=Z(s)$ for the p-port network using state equations.

transforms appear in eqns. (3-6) and (3-8), respectively. Taking the Laplace transform of eqn. (3-9) and (3-10), and solving for V(s), we get:

$$\underline{V}(s) = [\underset{\sim}{C}(s\underset{\sim}{I}-\underset{\sim}{A})^{-1}\underset{\sim}{B} + \underset{\sim}{D}]\,\underline{I}(s) \tag{3-11}$$

and, therefore, we get Z(s) to be

$$\underset{\sim}{Z}(s) \equiv [\underset{\sim}{C}(s\underset{\sim}{I}-\underset{\sim}{A})^{-1}\underset{\sim}{B} + \underset{\sim}{D}] \tag{3-12}$$

which is identically the inverse of the reduced node admittance matrix.

The matrix $(s\underset{\sim}{I}-\underset{\sim}{A})$ can be inverted by applying the similarity transformation as follows:

$$\underset{\sim}{A} = \underset{\sim}{M}\,\underset{\sim}{\Lambda}\,\underset{\sim}{M}^{-1} \text{ or } \underset{\sim}{\Lambda} = \underset{\sim}{M}^{-1}\,\underset{\sim}{A}\,\underset{\sim}{M}$$

$$\therefore \qquad \underset{\sim}{M}^{-1}(s\underset{\sim}{I}-\underset{\sim}{A})\underset{\sim}{M} = s\underset{\sim}{I} - \underset{\sim}{M}^{-1}\underset{\sim}{A}\underset{\sim}{M} = s\underset{\sim}{I} - \underset{\sim}{\Lambda}$$

$$\text{or} \qquad (s\underset{\sim}{I}-\underset{\sim}{A})^{-1} = \underset{\sim}{M}(s\underset{\sim}{I}-\underset{\sim}{\Lambda})^{-1}\underset{\sim}{M}^{-1} \tag{3-13}$$

where the inverse of $(s\underset{\sim}{I} - \underset{\sim}{\Lambda})$ is simply diag $\{(s-\lambda_1)^{-1},(s-\lambda_2)^{-1}, \ldots\}$ where λ_i are the eigenvalues* of the $\underset{\sim}{A}$ matrix and $\underset{\sim}{M}$ is the modal matrix. Substituting eqn. (3-13) into eqn. (3-12), we get,

$$\underset{\sim}{Z}(s) = [\underset{\sim}{C}\underset{\sim}{M}(sI-\Lambda)^{-1}\underset{\sim}{M}^{-1}\underset{\sim}{B} + \underset{\sim}{D}]$$

$$= [\underset{\sim}{\hat{C}}(sI-\Lambda)^{-1}\underset{\sim}{\hat{B}} + \underset{\sim}{D}] \tag{3-14}$$

where $\underset{\sim}{\hat{C}} \triangleq \underset{\sim}{C}\underset{\sim}{M}$ and $\hat{B} \triangleq \underset{\sim}{M}^{-1}B$. Equation (3-14) yields the entries of $\underset{\sim}{Z}(s)$ in partial fraction expansion form, which, as mentioned previously, is a more desirable form from a computational standpoint. All information about $\underset{\sim}{Z}(s)$

*Here we assume distinct eigenvalues; the repeated eigenvalues can be handled similarly.

is contained in the matrices $\hat{\underset{\sim}{C}}$, $\hat{\underset{\sim}{B}}$, $\underset{\sim}{D}$ and a vector containing the eigenvalues. An algorithm for implementing this approach is given in Fig. 3-2. It should be noted that the approach used here is completely numerical and does not involve any coding and decoding of symbols.

Now that an algorithm for obtaining the symbolic Z(s) is defined, we can make a comparison of the computational effort involved between using a symbolic inverse and the numerical inverse of the node admittance matrix at each frequency point.

The computational trade-off between the symbolic approach and a numerical approach for matrix inversion is very problem dependent. While a clear-cut winner cannot be established, a tentative answer can be obtained by noting the operations count, defined in terms of multiplications and additions, involved in the two schemes.

In the case of the numerical approach, the number of independent nodes, n, and the number of branches, b, are the most important quantities for determining the computational effort along with the number of frequency points at which the output is desired. Assuming that no sparse matrix techniques are used, the numerical inversion of an (nxn) matrix requires $O(n^3/3)$ units of work, where O( ) ≡ "order of", and 1 unit of work = one addition and one multiplication. For k frequency points, the work becomes $O(kn^3/3)$. This does not involve book-keeping and other pre- and post-processing steps such as pivoting and iterative refinement, which are usually necessary to insure reliability and robustness of the algorithm.

In the case of symbolic inversion using our approach, the important parameters in the computational effort are the dynamic degrees of freedom, d, and the number of ports, p, where voltages and currents are injected or measured. Using the QR algorithm [20,28] for computing the eigenvalues of

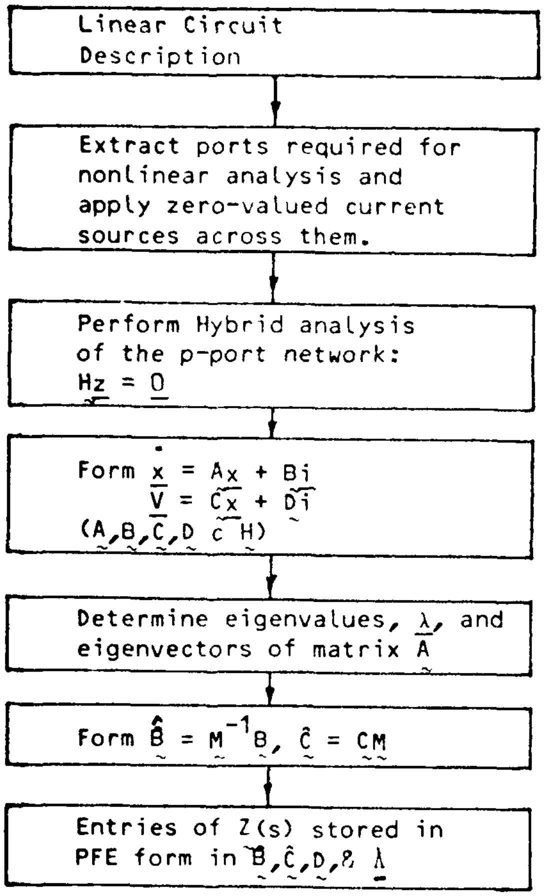


Fig. 3-2. Algorithm for inverting Y(s) symbolically.

the A matrix, the operation count is $O(8d^3)$. The total work required for obtaining the inverse at k frequency points is therefore $O(8d^3 + kdp^2)$. The number, p, depends on the number of nonlinearities in the circuit, and is usually small. Also, if the network complexity is less than the number of nodes, the symbolic approach would, in general, require less computational effort. As far as accuracy is concerned, both the QR algorithm and the Crout's algorithm with pivoting and iterative refinement yield accurate results.

The efficiency of the symbolic method rests heavily upon the availability on an efficient process for forming the state equations. The hybrid analysis method [19,20], which essentially reduces to the analysis of a resistive network, is well-suited for our purposes here.

3-4. Spectrum and Distortion Analysis Algorithm

The output spectrum and distortion indices for a nonlinear circuit with polynomial type nonlinearities can be computed on the basis of the material of Chapters 3 and 4. A flow-chart of the basic algorithm for such a computation is given in Fig. 3-3. We describe the steps involved in the following paragraphs:

Step 1: For the given nonlinear circuit, determine the dc operating point. Expand each nonlinear function into a Taylor series about the operating point to get a polynomial representation for the nonlinear element in terms of the incremental quantities. Thus, for example, a forward-biased diode having the "global" V-I representation

$$I = I_s[\exp(qV/nkT) - 1] \tag{3-15}$$

can be expanded into a Taylor series to yield the following incremental v-i representation:

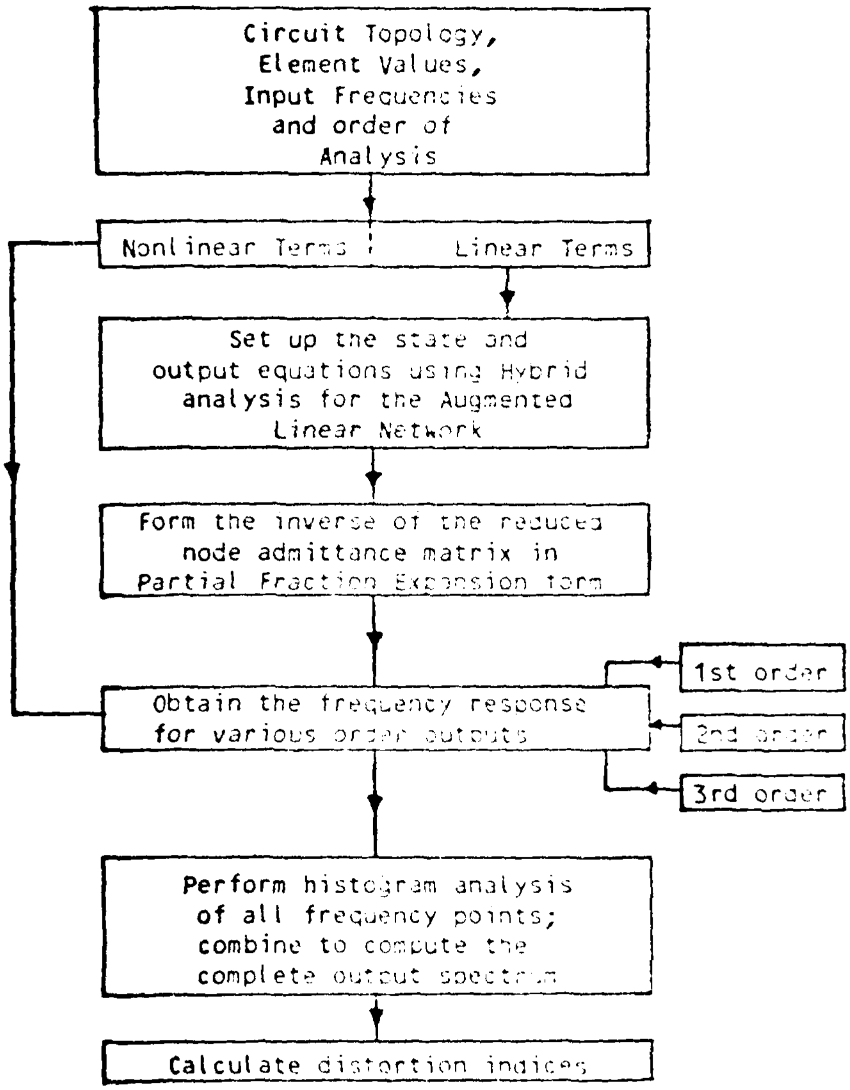


Figure 3-3. Algorithm for Spectrum and Distortion Analysis.

$$i = I_0 \frac{q}{mkT} v + \frac{I_0}{2!} \left(\frac{q}{nkT}\right)^2 v^2 + \frac{I_0}{3!} \left(\frac{q}{nkT}\right)^3 v^3 + \ldots \quad (3\text{-}16)$$

where I_0 is the dc operating current.

<u>Step 2</u>: Lump the linear part of the nonlinear elements with the existing linear network to form the augmented linear network. Extract as ports the nonlinear element branches and the branches that control the nonlinear element characteristics (dependent nonlinear element case), along with the output and source branches, from the augmented linear network. Let $\underline{V} = [V_1\ V_2\ \ldots\ V_p]$ and $\underline{I} = [I_1\ I_2\ \ldots\ I_p]$ denote the vector of voltages and currents for these ports, respectively.

<u>Step 3</u>: Using a symbolic analysis algorithm (see Fig. 3.2), obtain the entries of the Z matrix as a function of s, where

$$\underline{V}(s) = \underset{\sim}{Z}(s)\ I(s) \quad (3\text{-}17)$$

For each of the input sources, and their associated frequency tones, compute the first-order output voltages at each of the extracted ports by using the appropriate entries of the Z matrix. This step amounts to letting $s = j\omega_i$ in $z_{ij}(s)$, the entries of $\underline{Z}(s)$.

<u>Step 4</u>: The second-order output spectrum is evaluated using the following relationship:

$$\underline{V_2}(s_1,s_2) = \underset{\sim}{Z}(s_1+s_2)\ \underline{I_2}(s_1,s_2) \quad (3\text{-}18)$$

The vector $\underline{I_2}(s_1,s_2)$ is the second-order current source vector, which is formed by using the coefficients associated with the quadratic term of the nonlinear element and the first-order output at the controlling port(s) of the nonlinearity. The latter information was obtained in step 3. The given input tones are taken two at a time in eqn. (3-18), along with the information derived in Chapter 2, to evaluate the output voltages at each of the p-ports.

The third-order output spectrum is obtained in exactly the same manner. The first- and second-order outputs are used to form the third-order current source at each combination frequency, which is then pre-multiplied by evaluating $\underset{\sim}{Z}(s)$ at the combination frequency.

Step 5: Perform a histogram analysis of all frequency points and combine the responses at points which are repeated. The distortion indices are computed using:

$$HD_2 = \frac{|V_0(2\omega_i)|}{|V_0(\omega_i)|} \tag{3-19}$$

$$HD_3 = \frac{|V_0(3\omega_i)|}{|V_0(\omega_i)|} \tag{3-20}$$

where HD_2 and HD_3 denote the second and third order harmonic distortion indices.

3-5. Program PRANC.

The Program for Analysing Nonlinear Circuits, known as PRANC, is a digital computer program, written in FORTRAN IV, that computes up to the third-order complete output spectrum of a nonlinear circuit with polynomial nonlinearities driven by up to two multi-frequency inputs.* In the process it computes the Volterra transfer functions at each of the frequency combinations involved.

As mentioned previously, the solution of the nonlinear circuit problem reduces to the repeated solution of the linear circuit. To efficiently han-

*Thus, mixer-type circuits can be analyzed using PRANC.

dle this basic problem, PRANC uses a semi-symbolic approach [20] for analysing the augmented linear circuit. Specifically, the inverse of the reduced node admittance matrix is obtained in terms of the symbol s using the state equation formulation as described above.

The state equations for the linear circuit are formulated via the Hybrid analysis method [19,20]. If T denotes port branches in the tree [20] and C denotes port branches in the co-tree of a linear circuit, then the Hybrid analysis yields the following relationship:

$$\underbrace{\begin{bmatrix} H_{11} & H_{12} & H_{13} & H_{14} \\ H_{21} & H_{22} & H_{23} & H_{24} \\ H_{31} & H_{32} & H_{33} & H_{34} \\ H_{41} & H_{42} & H_{43} & H_{44} \end{bmatrix}}_{\underset{\sim}{H}} \underbrace{\begin{bmatrix} i_T \\ v_C \\ v_T \\ i_C \end{bmatrix}}_{\underline{z}} = \underline{0} \qquad (3\text{-}21)$$

By suitably forcing the various ports in the linear circuit into the tree and co-tree branches, PRANC uses the (3-21) formulation for setting up the state equations. All capacitor branches are extracted as ports which necessarily become part of the tree and all inductors, nonlinear element branches (which are assumed to be voltage controlled), and input and output branches, are extracted as ports which are forced as part of the co-tree. The matrix $\underset{\sim}{H}$ is obtained in a form where $\underset{\sim}{H}_{11} = \underset{\sim}{I}$ ($\underset{\sim}{I}$ being the identity matrix), $\underset{\sim}{H}_{12} = \underset{\sim}{H}_{21} = 0$, $\underset{\sim}{H}_{22} = \underset{\sim}{I}$. This yields the capacitor currents and the inductor and nonlinear element branch voltages in terms of known variables. Thus, the A, B, C, and D matrices in the state and output equations (see eqns. 3-9 through 3-12) are obtained from the submatrices of $\underset{\sim}{H}$. The formulation of eqn. (3-21) is quite fast, since it only involves the analysis of a resistive network.

It is noted that the matrix $\underset{\sim}{H}$ may not exist in idealized circuits. However, for most practical circuit this matrix is almost certain to exist [20]. It should also be noted that the above formulation of state equations tacitly assumes that no degenerate cutsets (all inductor-current source cutset) or degenerate loops (all capacitor-voltage source loop) are present in the linearized circuit. These restrictions are not very severe, especially when the realistic lossy models of circuit components are taken into account.

The next step in the PRANC algorithm is to determine the eigenvalues and the eigenvectors of the $\underset{\sim}{A}$ matrix. For this purpose, the double QR algorithm [28] for obtaining the eigenvalues is employed. The basic steps, such as matrix balancing, reduction to Hessenberg form, shift of origin, are included in this algorithm to make it efficient and reliable. The eigenvectors are also obtained in the process.

All information about the inverse of the reduced node admittance matrix is stored as three matrices and a vector. The matrices are $\hat{B}$, $\hat{C}$, and D (see eqns. 3-14), and the vector contains the eigenvalues. It is noted that the solution of eigenvectors for repeated eigenvalues can be a numerical unstable process [29]. Thus, the programs outputs a diagnostic message when such a case occurs.

The first-order voltage response at the prescribed ports is now computed from the entries of the open-circuit impedance matrix. These ports include: source port, output ports, nonlinear element ports, and ports which control the nonlinear element characteristics. The response is calculated for each user prescribed frequency, and stored as a two-dimensional array: port number vs. the frequency number.

The second-order voltage response is computed at each distinct combination of the input tones taken two at a time. The ports of interest are the same as that for the first-order response. The second-order current source vector, at a particular frequency combination, is formed by considering the nonlinear element type and the voltage(s) controlling it, which is determined from the first order response array. This vector is pre-multiplied by the open-circuit impedance matrix evaluated at the combination frequency to obtain the second-order transfer function vector at that frequency. The response voltage at this frequency is then determined from the transfer function value. The second-order transfer function values are again stored as a two-dimensional array: port number and the particular frequency combination.

The third-order response is determined similarly. The third-order current source vector is formed by properly picking out the values of the first- and second-order transfer functions. The indexing of the arrays is of critical importance to the efficient implementation of this scheme.

Since the hybrid analysis forms the basis for forming the open circuit impedance matrix, the following linear elements are allowed by the program*: resistors, capacitors, inductors, voltage or current sources, and all four types of controlled sources. The nonlinear elements are assumed to be voltage controlled, with the following polynomial descriptions:

$$i_p = a_1 f[v_p] + a_2 f[v_p^2] + a_3 f[v_p^3] \quad (3\text{-}22)$$

*A direct nodal analysis would only allow for voltage controlled current source.

$$i_p = a_{10}v_q + a_{01}v_r + a_{20}v_q^2 + a_{02}v_r^2 +$$

$$a_{11}v_qv_r + a_{30}v_q^3 + a_{03}v_r^3 + a_{12}v_qv_r^2 + a_{21}v_q^2v_r \qquad (3\text{-}23)$$

where i_n and v_n are currents and voltages across branch n, f is a linear operator of the type $\frac{d}{dt}$, $\int_{-\infty}^{t}$, or constant, and a_{ij} are constants. It should be noted that eqn. (3-23) models a 3-port device.

In the present version, PRANC imposes the following restrictions on the circuit parameters: maximum number of elements (both linear and nonlinear) = 60; maximum number of nonlinear elements = 10; maximum number of dependent nonlinear elements (eqn. 3-23) = 5; maximum number of reactive elements = 20; maximum number of independent nodes = 30; number of input frequencies = 5. These restrictions can be relaxed if desired. The modular structure and algorithms of PRANC makes it possible to extend the order of analysis in a straightforward manner. The limit on the highest order will eventually be dictated by the storage restrictions of the computer.

The validity of the results obtained from using PRANC has been verified through hand-worked examples and with the results obtained from using NCAP [24]. In Chapter 4 we present examples showing the results obtained from the use of PRANC.

CHAPTER 4

USER'S GUIDE FOR PRANC

4-1. Introduction

Based upon the theory of Chapter 2 and the algorithms of Chapter 3, PRANC (Program for Analyzing Nonlinear Circuits), a digital computer program, has been developed for the sinusoidal steady state analysis of circuits with multiple nonlinear elements and multiple multi-frequency input sources. The complete listing of the program is contained in Chapter 5.

The usefulness of PRANC is not restricted only to users who are well-versed in the Volterra series method; users with a basic knowledge of the significance of sinusoidal steady state analysis, eigenvalues (poles) of a linear system, and other related circuit analysis concepts can easily use the program, and understand the information provided by it. By suitably translating the circuit analysis problem into a prescribed sequence of well-defined statements - to be presented in this chapter - any user can use PRANC as an analysis tool.

To methodically and effectively use PRANC, the user is recommended to follow a three-step process:

1. Preliminary Data Preparation
2. Translation of Data for Analysis
3. Interpretation of Analysis' results.

The contents of this Chapter are organized on the basis of these steps.

In section 4-2, the considerations entailed in the preliminary data preparation are presented. The allowed elements, the user available options, and the program restrictions in terms of the circuit size and features are discussed.

Section 4-3 presents the sequence of input cards (input data to the program) for PRANC. The formats for each card in the sequence is described. The interpretation of the program output is the subject of section 4-4. Finally, several examples are presented in section 4-5 to illustrate the use of PRANC.

4-2. Preliminary Data Preparation

4-2.1 Allowable Element Types: The first step in any circuit analysis problem is the drawing of its complete circuit model. This diagram should include all elements which can be identified by PRANC.

The present version of PRANC is capable of identifying the following element types, which are depicted in Fig. 4-1:

- Independent voltage source
- Linear Components: Resistor, Inductor, and Capacitor
- Linear Dependent Sources: Voltage-controlled Voltage source, Current-Controlled Current-Source, Voltage-controlled current-source, and Current-controlled voltage-source.
- Nonlinear Components: Resistor, Inductor, and Capacitor
- Nonlinear Dependent Source: Voltage-controlled current-source.
- Bipolar Junction Transistor

The polarity convention assumed by PRANC is shown in Fig. 4-1. The current voltage relationships for linear elements are well-known. The nonlinear elements are assumed to be represented in the form of polynomials of branch voltage(s). Thus, PRANC handles nonlinear elements expressed as:

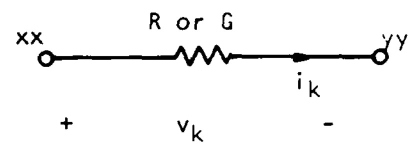
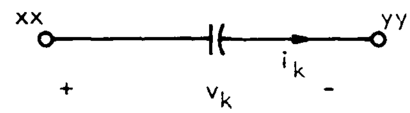
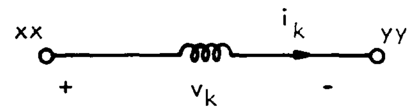
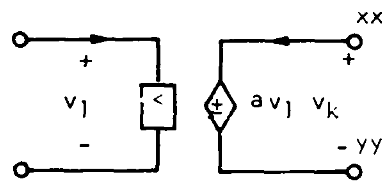
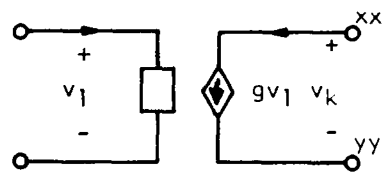
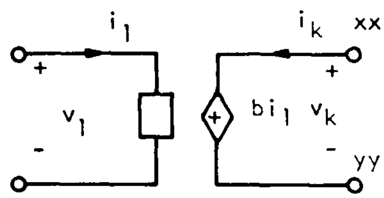
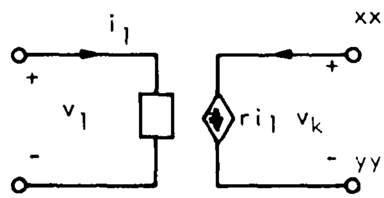
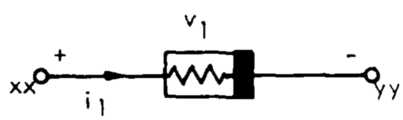
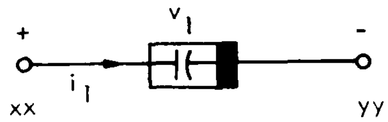
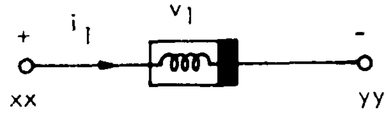
Element	Symbol
<u>Linear</u> <u>Components</u>: Resistor	
Capacitor	
Inductor	
<u>Linear</u> <u>Dependent</u> <u>Sources</u>: Voltage-Controlled Voltage Source	
Voltage-Controlled Current Source	
Current-Controlled Voltage Source	
Current-Controlled Voltage Source	
<u>Nonlinear</u> <u>Components</u>: Resistor	

Figure 4-1 PRANC Element Definitions

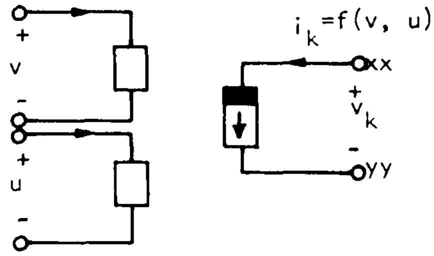
Capacitor



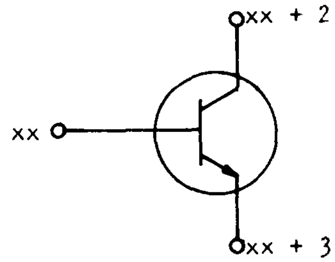
Inductor



Dependent Source:



Bipolar Junction Transistor



Input Source:

$$v_s(t) = \sum_k A_k \cos(\omega_k + \theta)$$

Z is R, L, or C

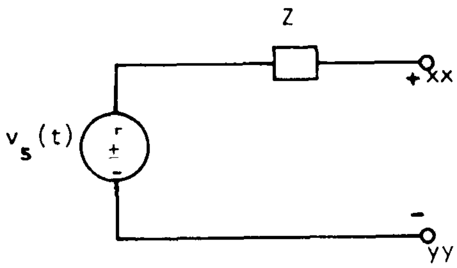


Figure 4-1 (Contd.) PRANC Element Definitions

$$i_{NL} = a_1 f[v] + a_2 f[v^2] + a_3 f[v^3] \tag{4-1}$$

or

$$i_{NL} = a_{10}v + a_{01}u + a_{20}v^2 + a_{02}u^2 + a_{11}vu$$

$$+ a_{30}v^3 + a_{03}u^3 + a_{21}v^2u + a_{12}vu^2 \tag{4-2}$$

where f is an operator of the form $\int$, $\frac{d}{dt}$, or a constant, u and v are branch voltages, and i_{NL} is the current across the nonlinear element. It should be noted that eqn. (4-1) is adequate to model a nonlinear capacitor, a nonlinear inductor, or a nonlinear resistor, and that eqn. (4-2) is suitable to model a 3-port or a 2-port voltage controlled nonlinear dependent source.

The representation of a nonlinear device in terms of a polynomial is covered in several papers and reports [7,10]. An example of the development of a polynomial representation for a semiconductor diode is given in Appendix A.

It should be noted that if a current-controlled nonlinear element is present in the circuit, the reversion of the series may be used. That is, given

$$v_{NL} = a_1 i_{NL} + a_2 i_{NL}^2 + a_3 i_{NL}^3 \quad (a_1 \neq 0) \tag{4-3}$$

We can express

$$i_{NL} = A_1 v_{NL} + A_2 v_{NL}^2 + A_3 v_{NL}^3 \tag{4-4}$$

where

$$A_1 = \frac{1}{a_1} \tag{4-5}$$

$$A_2 = -\frac{a_2}{a_1^3} \tag{4-6}$$

$$A_3 = \frac{1}{a_1^5}(2a_2^2 - a_1a_3) \tag{4-7}$$

where i_{NL} and v_{NL} are the current and voltage across the nonlinear element, respectively.

The element node numbers are shown by symbols xx and yy in Fig. 4-1. For devices representable in terms of a pair of nodes, or a collection thereof, the node numbers are assigned by the user. The node numbering for a bipolar junction transistor is done internally within the program once the node number for the base terminal of the transistor has been specified by the user. The model for the transistor used in PRANC is based on Narayanan's work [7], and is shown in Fig. 4-2 along with the program-assigned node numbers.

4-2.2. User Available Options:

PRANC performs the complete sinusoidal steady state analysis of a nonlinear circuit. In accomplishing this task, the program obtains the state equations and the eigenvalues for the linearized circuit, forms the entries of the open circuit impedance matrix* in partial fraction expansion form, and then computes the first-, second-, and third-order transfer function values at each combination of the positive and negative input frequency values. The output voltage at each frequency component is then computed

*This is an equivalent form of the inverse of the reduced node admittance.

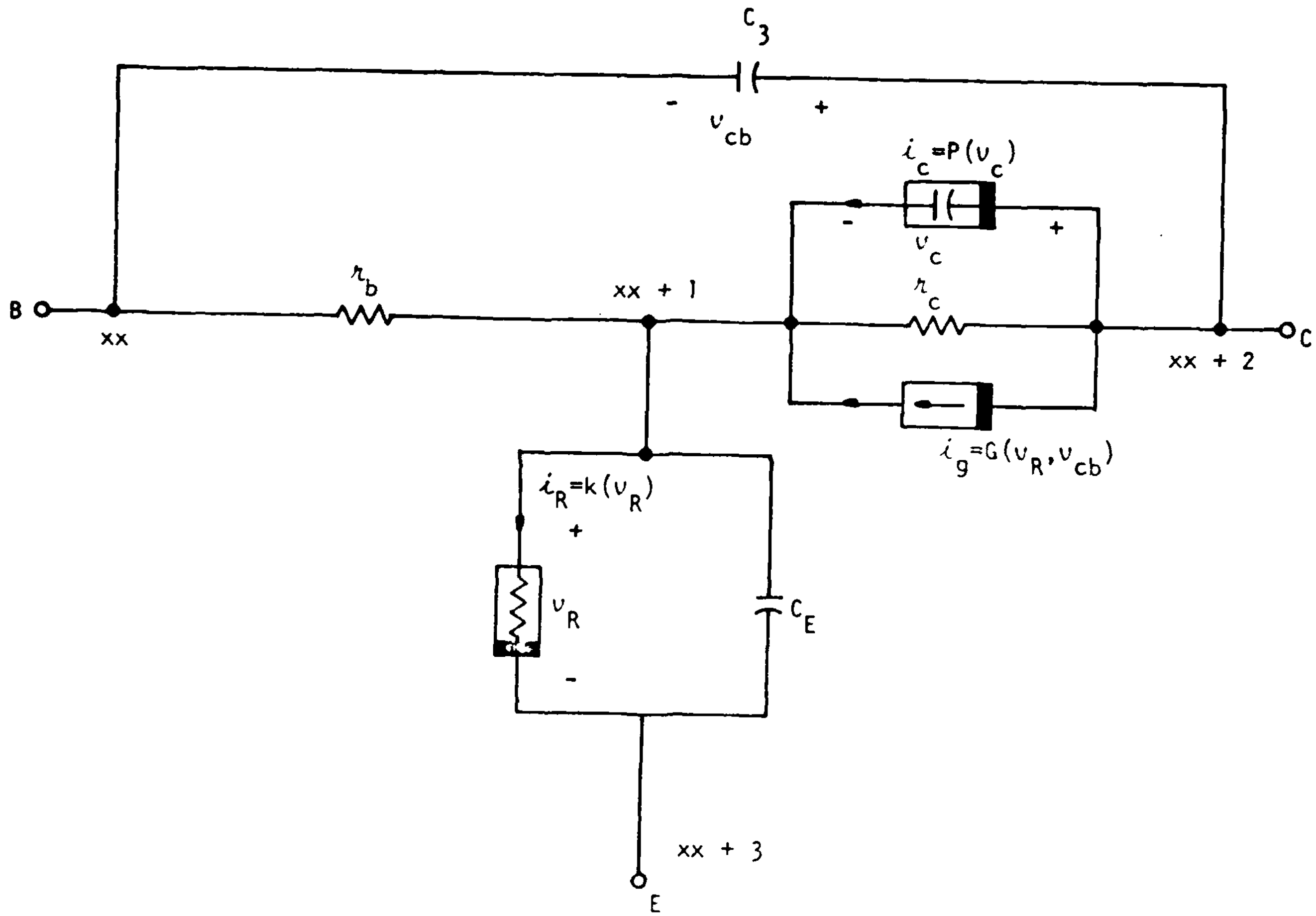


Figure 4-2. Transistor Equivalent Model.

from the transfer function. The sinusoidal steady state response is obtained after combining the various order responses at repeated frequency values.

In order to provide flexibility to the user to control the program output, several options have been incorporated in PRANC. These are described next.

1. <u>Frequency Sweep</u>. Many applications in distortion and spectrum analysis of nonlinear circuits calls for the study of the effect of frequency on the distortion products. A frequency sweep capability, which allows the user to request multiple analyses of a given circuit over a range of generator frequency values in a single execution, is provided by PRANC. This option can be called for by specifying the acronym FS on the option card.

PRANC allows the user to sweep up to five tones*. This allows the user to study the effect of frequency on the second- and third-order intermodulation products independently. In a third order analysis, a combination of three input frequencies is taken at a time to compute the amplitude of an intermodulation product; sweeping up to three frequencies is therefore sufficient to study the effect of frequency on an intermodulation product. Thus, given a fixed intermodulation frequence $\omega_{IM} = \omega_1 + \omega_2 - \omega_3$, where ω_1, ω_2, and ω_3 are the input frequencies, the effect of a change in the input frequencies on ω_{IM} can be investigated by simultaneously incrementing ω_1, ω_2, and ω_3 by a fixed amount across the band of interest. The study of the effect of frequency on a second-order distortion product is done similarly by varying two tones at a time. Both linear and logarithmic frequency sweeps are available on PRANC.

*Since PRANC generates the negative of the input frequencies internally, this is equivalent to sweeping ten frequencies.

2. Multiple Input Sources. Ordinarily PRANC assumes the nonlinear circuit to have a single multi-frequency input source. However, when a two source circuit, such as a mixer circuit, is to be analyzed, the acronym MX (mixer) on the option card can be used. PRANC will, in such a case, look for the description of the second input source. The first generator can have up to four input tones, and the second generator (the "local oscillator") only one input frequency.

3. Print and Plot Complete Spectrum. After computing the transfer functions and output voltages, PRANC performs a histogram type analysis of all frequency points to compute the complete output spectrum across a requested circuit element for printing and plotting purposes. Often times the user may be only interested in the transfer function and output voltage values, and may have no use for the complete output spectrum. In order to provide the flexibility for suppressing the printing and plotting of the complete output spectrum, an option to be specified by the user is available. By using the acronym PC on the option card the user can request for a print-out and plot of the complete output spectrum; an absence of PC on the option card signals the program to suppress the histogram analysis feature.

4. Output Port Print-out Selection. PRANC performs the analysis of the nonlinear circuit on a port basis. Two types of ports are extracted in the analysis: 1) Input and output ports specified by the user, and 2) controlling ports for the nonlinear elements. Depending on the number of nonlinear elements and the number of controlling voltage variables, the number of the extracted ports can become quite large and thereby result in an inordinately large amount of printed output if the transfer function and output voltage at each frequency component and at each of the ports is printed. To reduce

the amount of printed output, an option for the printing of selected output ports can be requested. By using the acronym AP (All extracted ports) on the option card, the transfer functions and output voltages at each of the extracted ports is printed; an absence of AP on the option card signals the program to print the transfer functions and output voltages at only the user-prescribed output ports.

5. State Space Description Print. The open circuit impedance matrix for the linearized circuit is obtained via the state space description (see eqn. 3-5). The user can request a print-out of this description by specifying the acronym SE on the option card. When SE is omitted from the option card, the printing of the $\underset{\sim}{A}$, $\underset{\sim}{B}$, $\underset{\sim}{C}$ and $\underset{\sim}{D}$ matrices is suppressed.

6. Eigenvalue, Modal Matrix Print. The eigenvalues or the poles of the linearized circuit, and their associated eigenvectors, are computed by PRANC. The user may access this information by specifying NM (natural modes) on the option card. The eigenvalues and the modal matrix are not printed when the letters NM are omitted from the option card.

7. Open Circuit Impedance Matrix Print. The open circuit impedance matrix for the linearized circuit is obtained in partial fraction expansion form by PRANC. Each entry of this matrix is obtained in terms of a set of pole-residue pairs and can be written as:

$$z_{ij}(s) = \sum_k \frac{r_k}{s-\lambda_k} + \text{constant} \qquad (4\text{-}8)$$

where r_k is the residue associated with the pole λ_k. Knowing all the entries of the open-circuit impedance matrix in the form (4-8), it is possible to obtain the higher order transfer functions in terms of s_i [23]. By using the acronym PR (pole-residue) on the option card, all information required to obtain each entry of the open circuit impedance matrix in the form (4-8) can be accessed from PRANC.

8. Debug Print. The hybrid analysis formulation is used by PRANC to set up the state space description of the linearized circuit. All important intermediate results leading to the determining of the hybrid matrix can be obtained by the user by requesting a debug run. This option is invoked by specifying the acronym DB on the option card.

4-2.3. Program Restrictions

The present version of PRANC imposes the following restrictions on the circuit size:

Maximum number of elements (both linear and nonlinear) =	60
Maximum number of nonlinear elements[+] =	10
Maximum number of dependent nonlinear elements =	5
Maximum number of reactive elements =	20
Maximum number of independent nodes =	30
Maximum number of input frequencies* =	5
Maximum number of extracted ports** =	25
Maximum number of inputs =	2

In addition to the above size restrictions, there are other restrictions imposed by the algorithms used: the presence of degenerate (all capacitor-voltage source) loops or degenerate (all inductor-current source)

[+]A bipolar transistor accounts for three nonlinear elements.
*These are the sine wave input frequencies. The negative frequencies are generated within the program.
**Number of extracted ports $\leq$ NO + NINL + 3NDNL + 1;
NO $\equiv$ number of requested outputs
NINL = number of independent nonlinear elements
NDNL = number of dependent nonlinear elements.

cutsets [20] will lead to erroneous results. It should be noted that this restriction is not severe when the realistic lossy models for capacitors or inductors are used. A series resistance with a capacitor or a shunt resistance with an inductor to account for the element non-idealities will insure the absence of any of the aforementioned degenerate cases.

Another restriction encountered in PRANC is related to the determination of the eigenvectors. It is well known [29] that the computation of the eigenvectors for repeated eigenvalues can be an ill-conditioned problem. Thus, whenever a linearized circuit has repeated eigenvalues, PRANC outputs a diagnostic message*. Again it is remarked that this restriction is not very severe. One can easily concoct simple network examples with repeated eigenvalues; but in real-life circuits, the probability of encountering repeated eigenvalues is very low - particularly when one considers the method of storing numbers in the finite length word of any digital computer.

To summarize this sub-section, the following three-step procedure is recommended to the user as part of the preliminary data preparation:

Step 1: Examine the circuit under consideration to insure that all elements are recognizable by PRANC. Furthermore, insure that there are no degenerate loops or degenerate cutsets [20]. If such conditions exist, the following remedy is recommended: place a negligibly small resistor in a degenerate loop; place a negligibly small conductance in parallel with one of the elements of the degenerate cutset.

Step 2: Assign consecutive numbers to all elements in the circuit (including a bipolar transistor) from 1 to NB and all nodes in the circuit from 0 to NN, where NB is the number of elements (both linear and nonlinear) and NN is

*It should be noted that this is due only to the numerical problems and that the theory of chapters 2 and 3 is still valid.

the number of independent nodes. Node number 0 is assumed to be the ground node. Insure that the circuit size does not exceed the limits imposed by the present version of PRANC.

Step 3: Note the number of linear and nonlinear elements, and the number and unit of the input frequencies. Based on the list of available options, select the ones desirable for the circuit analysis problem at hand.

4-3. Input Description for PRANC

In this sub-section, the details of the prescribed sequence of cards needed for using PRANC are presented. After the preliminary data preparation step is done, the procedure for translating the circuit description into input data is straightforward.

Assuming that PRANC is stored in the computer, the sequence of cards needed for the analysis of a nonlinear circuit with a single source is shown in Fig. 4-3; the case of the two-input source circuit is shown in Fig. 4-4. There are basically six types of cards present in the input data for PRANC. These are: 1) Title card; 2) Option Card; 3) Analysis Parameter card; 4) Linear Component description cards; 5) Nonlinear component description cards; and 6) Generator description cards. The details of the contents of each of these card types are described next.

1. Title Card: This card is read in with an 80A1 form and is reproduced as the first line of the output.

2. Options Card: This card tells the program which options, described in section 4-2.2, are desired by the user. Each option has a two-letter acronym associated with it. These acronyms are summarized in Table 4-1. Starting in column 1 the user must punch a contiguous string of the acronyms re-

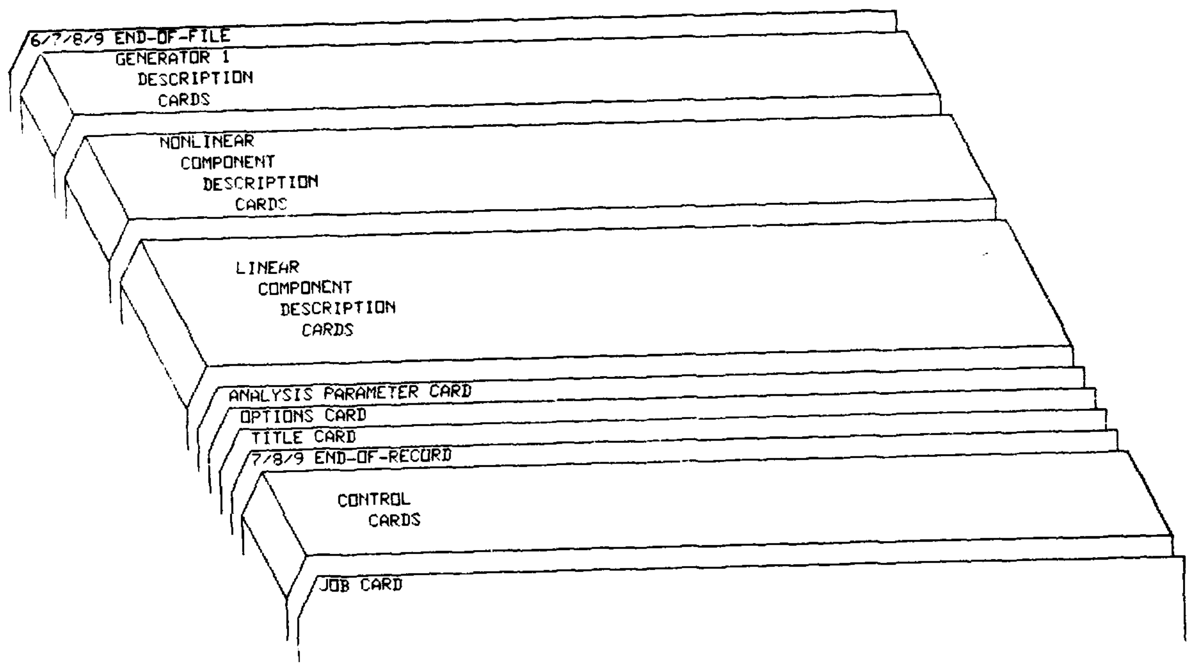


Figure 4-3. Sequence of Cards for Single Input Circuits

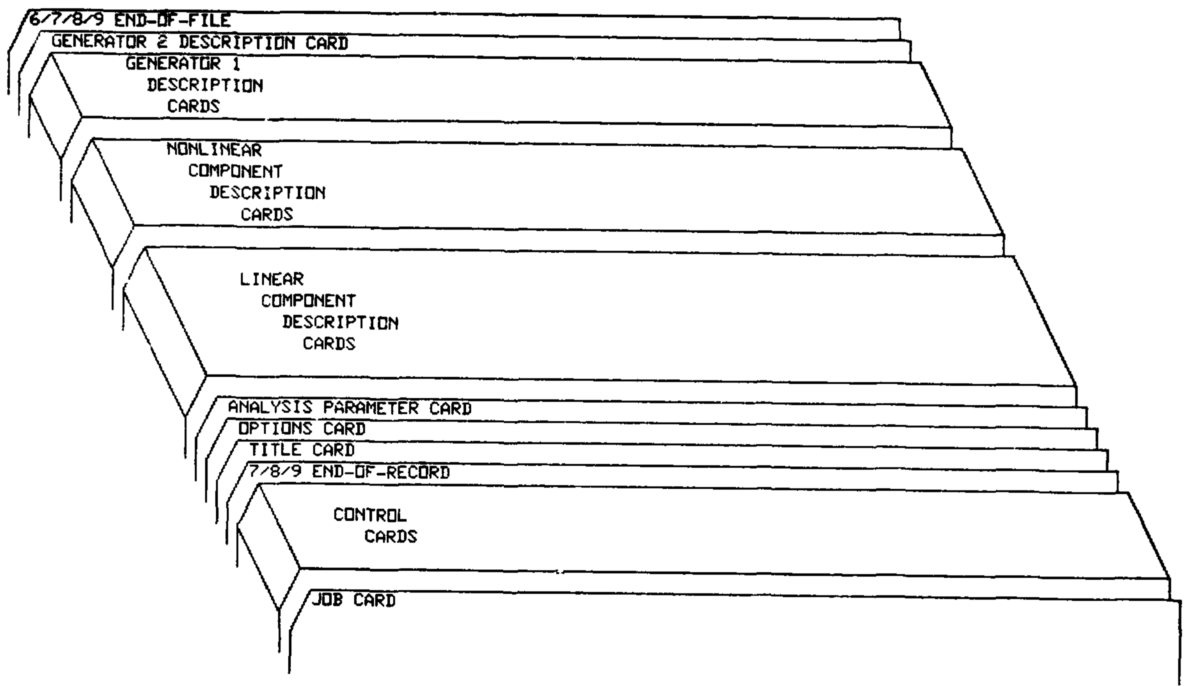


Figure 4-4. Sequence of Cards for Two-Input Circuits

quired to request the specific options. The card must therefore be in the following format:

Column	Format	Description
1-2	A2	First desired option acronym
3-4	A2	Second desired option acronym
5-6	A2	Third desired option acronym
.	.	
.	.	
.	.	

See section 4-5 for examples.

3. Analysis Parameter Card: The analysis parameter card contains information regarding the number of linear elements, the number of nonlinear elements (the transistor should be counted as 1 nonlinear element by the user), the number of sinusoidal frequencies (≤ 5) in the input signal*, the unit of the input frequencies, and the type of frequency sweep (if desired). This card must be in the following format:

Column	Format	Description
1-2	I2	No. of linear elements
3-4	I2	No. of nonlinear elements
5	I1	No. of input frequencies
6-8	A3	Unit for the input frequencies use RAD for rad/sec. Hz for Hertz
9-11	A3	Type of frequency sweep, if desired; LIN for linear and LOG for logarithmic

*This does not include the local oscillator frequency in the case of a mixer circuit.

Table 4-1. Summary of Available User Options on PRANC

	Option Acronym	Option Description
1.	AP	Print all extracted ports output information
2.	DB	Debug for Hybrid analysis: print all intermediate results
3.	FS	Frequency sweep capability
4.	MX	Two-input circuit for analysis
5.	NM	Print eigenvalue and modal matrix information
6.	PC	Print and plot complete output spectrum
7.	PR	Print pole-residue information for the open-circuit impedance matrix
8.	SE	Print state-space description of the linearized circuit

It should be noted that the I-format and the A-format are always right justified.

4. Linear Component Description Cards: Each branch containing a linear element save the independent source(s) and its impedance(s) must be described in terms of its topological connections, its element value and type, and its controlling branch number, if any. Each linear element description card must use the following format:

Column	Format	Description
1-3	I3	Branch number
4-6	I3	Positive ("From") node number
7-9	I3	Negative ("To") node number (Sign convention for PRANC is shown in Fig. 4-1)
10-11	A2	Element type. The following element types and their acronyms are recognized by PRANC: R Resistance G Conductance L Inductance C Capacitance CV Current-controlled Voltage source VV Voltage-controlled Voltage source VC Voltage-controlled Current source CC Current-controlled Current source (Note: R,L,G, or C must be present in column 11: right-justified)
12-21	E10.3*	Element value of R,G,L,C, or dependent source.
22-24	I3	Branch number for the controlling branch of CC, CV, VV, or VC. For other element

*(Note: For an E-format input: the exponent appears as a signed two digit integer in the three right most columns of the format field, and is preceeded by a letter E, which is preceeded by the floating point value. Thus, for example, a 6.6µF capacitor value should appear as:

12	13	14	15	16	17	18	19	20	21
	6	.	6	0	0	E	-	0	6

).

		types this should be left blank.
25	I1	This column is used to provide multiple output capability. A 1 in this column indicates that the current branch number is an output branch; a blank indicates otherwise.
26	A1	An asterisk (*) in this column indicates that the complete output spectrum across this branch should be printed and plotted; a blank indicates otherwise. Note: only one such branch is allowed in the current version of PRANC.

5. Nonlinear Component Description Cards: Two cards are used to describe each nonlinear capacitor, inductor, resistor, or a dependent source. The first card describes the nonlinear component type and its connection in the circuit; and the following card, the second card, defines the coefficient values in the polynomial expansion of the nonlinear element (as per eqns. 4-1 and 4-2). The nonlinear components are assumed to have a voltage-controlled current.

The format for the two cards required to define a two terminal nonlinear component is as follows:

First Card:

Column	Format	Description
1-3	I3	Component number
4-6	I3	Positive ("from") node number of the component
7-9	I3	Negative ("to") node number of the component
10-11	A2	Element Type. The following acronyms are allowed for the various element types: NC Nonlinear Capacitor NL Nonlinear Inductor NR Nonlinear Resistor ND Dependent Nonlinearity (eqn. 4-2)
12-14	I3	First controlling voltage branch

		number for the dependent nonlinearity. These columns are left blank in the case of NC, NL, or NR.
15-17	I3	Second controlling voltage branch number for the dependent nonlinearity. These columns are left blank in the case of NC, NL, NR, or single-voltage-controlled dependent nonlinearity.

<u>Second Card</u>: This card is used to define the coefficients of the polynomial describing the nonlinear element described on the first card. The format for this card is:

Column	Format	Description
1-10	E10.3	Coefficient a_1 in eqn. (4-1) or coefficient a_{10} in eqn. (4-2)
11-20	E10.3	Coefficient a_2 in eqn. (4-1) or coefficient a_{01} in eqn. (4-2)
21-30	E10.3	Coefficient a_3 in eqn. (4-1) or coefficient a_{20} in eqn. (4-2)
31-40	E10.3	Coefficient a_{02} in eqn. (4-2)
41-50	E10.3	Coefficient a_{11} in eqn. (4-2)
51-60	E10.3	Coefficient a_{30} in eqn. (4-2)
61-70	E10.3	Coefficient a_{03} in eqn. (4-2)
71-80	E10.3	Coefficient a_{21} in eqn. (4-2)
1-10 (new card)	E10.3	Coefficient a_{12} in eqn. (4-2)

Three cards are needed to describe each bipolar transistor in the circuit. The first card indicates that a transistor is present and also specifies the node number for the external base terminal. The second and third cards input the transistor parameters for the purpose of PRANC modelling. These parameters include the following (see Fig. 4-2):

Parameter No.	Parameter Name	Description
1	n	Avalanche Exponent

2	V_{cB}	Collector-base bias Voltage
3	V_{cBO}	Avalanche Voltage
4	μ	Collector Capacitance Exponent
5	I_c	Collector bias current
6	I_{cmax}	Collector current at maximum d.c. current gain
7	a	h_{FE} nonlinearity coefficient
8	h_{FEmax}	maximum d.c. current gain
9	k	collector capacitance scale factor
10	Ref	Diode non-ideality factor
11	C_{je}	Base-emitter junction space charge capacitance
12	C_2'	Derivative of base-emitter diffusion capacitance
13	r_b	Base resistance
14	r_c	Collector resistance
15	C_1	Base-emitter capacitance
16	C_3	Base-collector and overlap capacitance

Once the external base terminal node, xx, has been specified by the user, the following node numbers are internally assigned by the program to the other terminals in the transistor model:

xx + 1 : Internal Junction

xx + 2 : External collector

xx + 3 : External emitter

The user must therefore take caution in not assigning these node numbers elsewhere in the circuit.

For the bipolar junction transistor description, the following sequence of cards are used:

First Card: The format of the first card for the description of a BJT is identical to that for the two terminal nonlinear components. Accordingly, the following format is used:

Columns	Format	Description
1-3	I3	Component number*
4-6	I3	External base node number
7-9	I3	(blank)
10-11	A2	The acronym TR in these columns signals the presence of a bipolar junction transistor.

A TR in columns 10-11 on the first card of the nonlinear description cards causes PRANC to read two additional cards describing the transistor parameters. The format and the order in which the parameters are read is as follows:

Second Card: Columns	Format	Description	
1-10	E10.3	n	: Avalanche Exponent Value
11-20	E10.3	V_{cB}	: Collector-base bias voltage value
21-30	E10.3	V_{cBO}	: Avalanche voltage value
31-40	E10.3	μ	: Collector capacitance exponent value
41-50	E10.3	I_c	: Collector bias current value
51-60	E10.3	I_{cmax}	: Collector current at maximum d.c. gain value
61-70	E10.3	a	: h_{FE} nonlinearity coefficient value
71-80	E10.3	h_{FEmax}	: maximum current gain value

*Each transistor should be counted as one nonlinear component in the circuit.

Third Card:

Columns	Format		Description
1-10	E10.3	k	: collector capacitance scale factor value
11-20	E10.3	Ref	: diode non-ideality factor value
21-30	E10.3	C_{je}	: Base-emitter junction space-charge capacitance value
31-40	E10.3	C_2'	: Derivative of base-emitter diffusion capacitance value
41-50	E10.3	r_b	: base resistance value
51-60	E10.3	r_c	: collector resistance value
61-70	E10.3	C_1	: base-emitter capacitance value
71-80	E10.3	C_3	: base-collector and overlap capacitance value

In summary, the nonlinear component description cards are a sequence of cards where:

1) Two cards are used to describe each nonlinear resistor, nonlinear capacitor, or nonlinear inductor;

2) Three cards are used to describe each nonlinear dependent source;

3) Three cards are used to describe each bipolar junction transistor in the circuit.

6. Generator Description Cards: PRANC assumes the independent source to be a voltage source in series with an impedance, as shown previously in Fig. 4.1. The impedance can be a linear resistor, a linear capacitor, or a linear inductor. Two types of cards are required to describe the generator: the first card specifies the generator connection in the circuit and the succeeding cards describe the frequencies and their associated amplitudes along with the parameters for frequency sweep capability, if desired by the user.

Only two nodes are needed to specify the connection of the generator to the circuit.

The input voltage source is assumed to have the following form:

$$v_s(t) = \sum_{i=1}^{n} A_i \cos(\omega_i t + \theta_i);\ n \leq 5 \tag{4-9}$$

The user is therefore required to input the values for A_i, ω_i, and θ_i to describe the input source.

When the frequency sweep capability is requested by the user on the option card, the following three quantities must also be specified along with A_i, ω_i, and θ_i: 1) the number of steps or frequency increments; 2) the highest or terminal value of the frequency sweep; and 3) type of the desired sweep, which indicates whether the increment is to be linear (additive) or logarithmic (multiplicative).

It should be noted that the number of steps defines the number of times the circuit is to be analyzed. For linear sweeps the value of the increment is calculated by the program according to the expression:

$$INC_i = \frac{HFR_i - FR_i}{NSTP_i - 1} \tag{4-8}$$

where INC_i ≡ frequency increment value for the i-th frequency,

HFR_i ≡ highest value for the i-th frequency,

FR_i ≡ starting value for the i-th frequency,

$NSTP_i$ ≡ number of increments for the i-th frequency,

Similarly, for a logarithmic sweep, the increments are calculated as follows:

$$INC_i = \left[\frac{HFR_i}{FR_i}\right]^{\frac{1}{NSTP_i - 1}} \qquad (4\text{-}11)$$

(4-12)

In determining the value for the number of increments, the user should be aware that the highest and the starting frequency values each count as an increment. It should also be noted that multiple frequency sweep specifications always result in simultaneous increments of the frequency values involved. The largest defined "number of increments" value determines the number of analyses to be performed in such cases. As the analysis progresses, each frequency value will be incremented until its highest value has been reached, after which it will remain constant until all defined frequency sweeps have been satisfied.

The first card in the generator description card has the following input format:

Column	Format	Description
1-3	I3	Positive ("from") node number for the generator
4-6	I3	Negative ("To") node of the generator
7-8	A2	Source Impedance Type: _R, _L, or _C
9-18	E10.3	Source impedance element value

The cards following the above card provide information about each frequency value, along with its associated amplitude and phase, and its frequency sweep parameters. The format used to describe the i-th input frequency is as follows:

Column	Format	Description
1-10	E10.3	Amplitude value for the i-th frequency
11-20	E10.3	i-th input frequency value (must be greater than 0)
21-30	E10.3	Phase value in degrees for the i-th frequency
31-40	E10.3	Highest value for the i-th input frequency. Should be left blank when frequency sweep capability is not desired.
41-42	I2	Number of increments desired for the i-th frequency.

When two input sources are present in the circuit being analyzed, and the acronym MX has been included on the option card, the card immediately following the above "frequency description" cards is used to define the second source ("local oscillator") parameters. The second source is again assumed to be a voltage source with a series impedance of the resistive, inductive, or capacitive type. Only one frequency value is however allowed for the second source. The description of the second source must have the following format:

Column	Format	Description
1-3	I3	Positive ("from") node number for the source
4-6	I3	Negative ("To") node number for the source
7-8	A2	Source impedance type: R, L, or C
9-18	E10.3	Source impedance element value
19-28	E10.3	Amplitude value of source
29-38	E10.3	Source ("local oscillator") frequency value
39-48	E10.3	Phase value in degrees for the source

In section 4-5 we shall present concrete examples to illustrate the typical sequence of cards used to translate nonlinear circuit problems for analysis using PRANC.

4-4. Interpretation of PRANC Output

A typical PRANC output comprises a large volume of printed information. In general, even when all user available options are suppressed, the output consists of: 1) images of all input cards; 2) all circuit devices* with their associated parameters and polynomial representation of their nonlinearities; 3) the description of the augmented linear network; 4) the description of all extracted ports; and 5) the transfer functions and output voltages across the desired output ports. The transfer functions and the output voltages are printed for all non-negative ("positive" frequency spectrum) combinations of every positive and negative input sinusoidal frequencies**, in both cartesian and log polar form. Thus, if a two-tone generator is specified by the user, with $2f_2 > f_1 > f_2$, PRANC will print the transfer function and output voltage values at the following frequency combinations:

First order : f_1, f_2

Second order : $2f_1, f_1+f_2, f_1-f_2, 0, 2f_2$

Third order : $3f_1, 2f_1+f_2, f_1, 2f_1-f_2, f_1+2f_2, 3f_2, 2f_2-f_1, f_2$

When the available user options are used, additional information about the circuit is provided by PRANC. The details of each of the available option was presented in section 4-2. We briefly repeat their functions here.

*These include bipolar junction transistor parameters in the present version.
**The user specifies only the positive sinusoidal frequencies; PRANC generates their negative values within the program.

When the acronym SE is punched on the option card, PRANC will print the complete state space formulation for the augmented linear network. It is well-known [20] that, like the nodal or loop analysis, a linear network is completely characterized by its state space description. By isolating the dynamic (energy storage) elements in the linear network, the state equation description emphasizes the dynamic character of the linear part of the non-linear circuit under study. PRANC isolates the capacitor voltages and the inductor currents as the state variables for the linearized network. It prints the A, B, C, and D matrices of the following vector equations:

$$\begin{aligned} \dot{\underline{x}} &= \underset{\sim}{A}\underline{x} + \underset{\sim}{B}\underline{i} \\ \underline{y} &= \underset{\sim}{C}\underline{x} + \underset{\sim}{D}\underline{i} \end{aligned} \qquad (4\text{-}13)$$

where the vector $\underline{x} = [v_{c1}\ v_{c2} \ldots v_{cn} \vdots i_{L1}\ i_{L2} \ldots i_{Lk}]^T$,

the vector $\underline{i} = [i_1\ i_2 \ldots i_p]^T$,

and the vector $\underline{y} = [v_1\ v_2 \ldots v_p]^T$.

Here v_{ci} is the i-the capacitor voltage, i_{Li} is the i-th inductor current, and v_k and i_k are the voltages and currents for the k-th extracted port, respectively. The order in which the states are arranged is identical to the order in which the capacitors and inductors appear in the augmented linear description, which is always printed by PRANC in a typical successful execution of the program.

When the acronym NM appears on the option card, the eigenvalues (poles) and the modal matrix for the augmented linear network is printed. The significance of this information is well-known [20]: the poles have a direct bearing on the linear system response and stability; the modal matrix can be used to study the zero-input response along with the observability and controllability properties of the linearized system.

AD-A088 422 PURDUE UNIV LAFAYETTE IND F/G 9/3
PRANC: PROGRAM FOR ANALYZING NONLINEAR CIRCUITS.(U)
MAY 80 H K THAPAR, B J LEON F30602-78-C-0102
UNCLASSIFIED RADC-TR-80-139 NL

2 OF 3
AD A
088422

The presence of the acronym PR on the option card causes PRANC to print the pole-residue information of each entry of the open-circuit impedance matrix for the p-port augmented linear circuit. This information can be used to construct the higher order transfer functions in terms of the transform variables s_i. Multi-dimensional transform theory [5] can then be applied to these transfer functions to get more insight into the operation of the nonlinear circuit.

The presence of the acronym AP on the option card causes PRANC to print the transfer function and the output voltage values for all the ports extracted for analyzing the nonlinear circuit problem. These ports include: 1) input source port(s); 2) user requested output ports; 3) the ports at which the nonlinear elements are present; and 4) the ports which control the nonlinear element characteristics.

When the acronym PC is present on the options card, the complete steady-state response at the "most desirable", user-specified output port is obtained and printed. The logarithm of the output voltage is also plotted as a bar-graph, which has the same display characteristic as a spectrum analyzer. As mentioned previously, frequency components appearing in the first-order response may appear in higher-order responses also. The function of the option under consideration is to combine these responses and print the response at only the set of distinct frequencies.

The use of the debug option (DB) causes PRANC to print the intermediate results of hybrid analysis of the augmented linear circuit. This option has been incorporated for the debugging of the linear circuit analysis and is not recommended for use during a typical run. An understanding of hybrid analysis [20] is necessary to interpret the output -- which can be quite voluminous - from the debug run.

4-5. Examples using PRANC

A set of examples are presented in this section to illustrate the use of PRANC for obtaining the steady-state response of nonlinear circuits. Each example will contain the problem statement, the sequence of punched data cards, the computer printed output, and some remarks on the printed output.

Example 4-1: Single Stage Untuned Amplifier Circuit

Consider the untuned, bipolar transistor amplifier of Fig. 4-5. The input source comprises of three frequencies. The sequence of data cards used are shown in Fig. 4-6 and Fig. 4-7. Note that the second card in the sequence, referred to as the option card previously, calls for the pole-residue information (acronym PR) and the printing and plotting of the complete output spectrum (acronym PC) across the 50-ohm resistor present between node 6 and the ground node (card no. 9). By not including AP on the option card, the printing of the responses at all extracted ports was suppressed; instead, only the responses across the 50-ohm resistor and the 0.1 ohm resistor are printed. The transistor parameters used in the example are listed on the computer print-out.

Referring to the computer printed output, we note that all the user-specified information has been listed. A description of the augmented linear network, which is formed after the linear parts of the nonlinear elements have been lumped with the existing linear network, is also listed. In the present examples, six ports were extracted as shown by the port assignment description. The open-circuit impedance matrix is therefore of dimension 6x6. The pole-residue information (see eqn. 4-8) for each of the entries of this matrix is also provided. The transfer function and the output voltage values for the various orders and frequency combinations have also

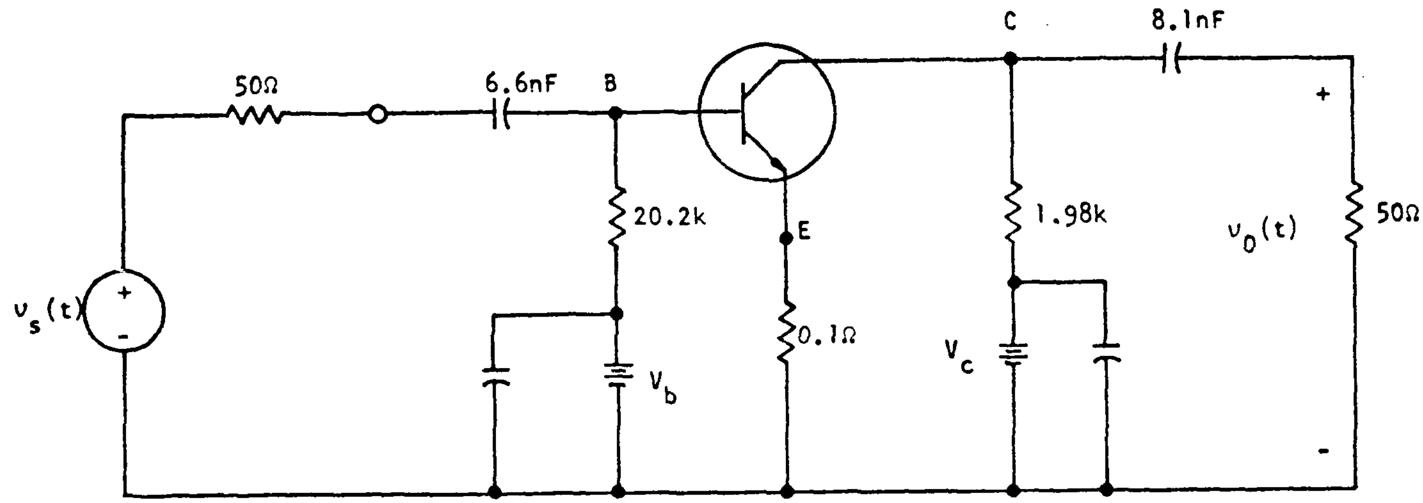


Figure 4-5. A Transistor Amplifier Circuit.

been listed. Finally, the output spectrum across port 3 (node pair 6-0) has been printed and plotted. The total execution time for this example on the CDC 6500 computer is approximately 4.8 seconds.

```
EXAMPLE 4-1: SINGLE STAGE UNTUNED AMPLIFIER CIRCUIT
[illegible]
[illegible]13 [illegible]Z
[illegible]1[illegible]1[illegible]   C6.6[illegible]E-09
[illegible]                R[illegible]E+0[illegible]
[illegible]                R[illegible].1[illegible]            -1       (output [illegible])
[illegible]4[illegible]    R1.98[illegible]E+05
[illegible]4[illegible]    C[illegible].1[illegible]E-0[illegible]
[illegible]                R[illegible]                   1*       (Output [illegible] complete output [illegible] desired [illegible])
[illegible]8[illegible]    C4.[illegible]E-12
[illegible]       TR
[illegible]          9.[illegible]      140.00      0.548      10.000E 0310.00E 030.120      8.200
[illegible]       121.00      0.5400E-090.5900E-0710.100      635.00E+030.000      1.0000E-12
001000 R50-00
1.0000E+000.000      2.5000E+06
1.0000E+000.000      3.0000E+06
1.0000E+000.000      3.5000E+06
```

Fig. 4-6. Data Preparation for Example 4-1

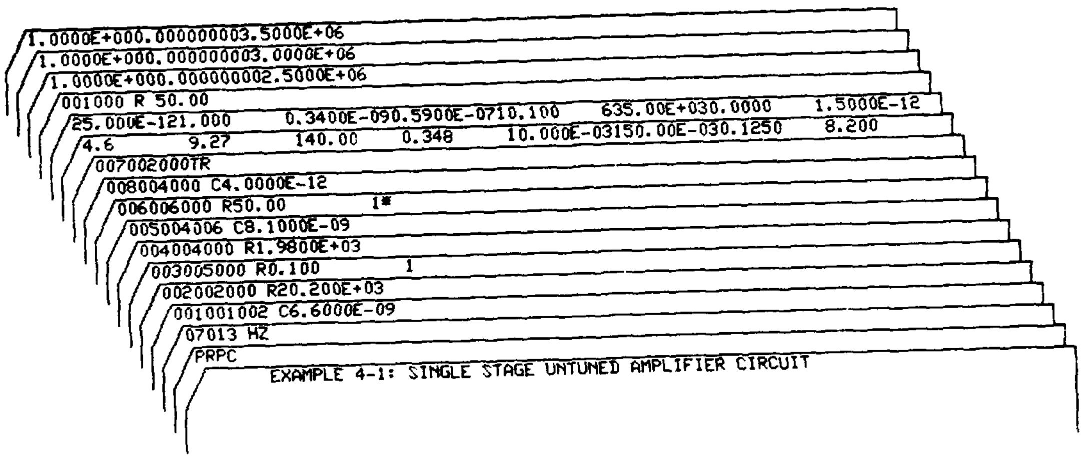


Figure 4-7. Data Cards for Example 4-1

EXAMPLE 4-1: SINGLE STAGE UNTUNED AMPLIFIER CIRCUIT

USER REQUESTED OPTIONS:
DEBUG PRINT-OUT: NO
FREQUENCY SWEEP CAPABILITY: NO
TWO-INPUT CIRCUIT: NO
STATE EQUATION PRINT-OUT: NO
EIGENVALUES MODAL MATRIX PRINT-OUT: NO
OPEN-CIRCUIT IMPEDANCE MATRIX PRINT-OUT: YES
COMPLETE OUTPUT SPECTRUM PLOT: YES
ALL EXTRACTED PORT OUTPUTS: NO

NETWORK DESCRIPTION:
....

LINEAR ELEMENTS

BRANCH NUMBER	FROM NODE	TO NODE	ELEMENT TYPE	ELEMENT VALUE	CONTROL BRANCH
1	1	2	C	3.300E-03	
2	2	0	R	2.020E+04	
3	5	0	R	1.000E-01	
4	4	0	R	1.030E+03	
5	4	0	C	3.100E-03	
6	3	0	R	5.000E+01	
7	4	0	C	4.000E-12	

TRANSISTOR PARAMETERS:

N= 4.300	VCE= 6.270	VCBO=140.000	MU= .348
IC= 1.000E-02	ICMAX= 1.500E-01	A= .125	HFEMAX= 8.20
K= 2.500E-11	REF= 1.70	CJE= 3.400E-10	CX2= 5.800E-03
RB= 10.100	RC= 3.350E+05	C1= 0	C3= 1.500E-12

NONLINEAR ELEMENTS

FROM NODE	TO NODE	TYPE	CONTROL (1)	CONTROL (2)	POLYNOMIAL COEFFICIENTS		
3	5	NR			A1= 4.0021E-01	A2= 3.2083E+00	A3= 1.0371E+02
4	2	NO	14	13	A10= 3.3434E-01	A01= 1.8708E-08	A20= 7.3458E+00
					A02= 3.3331E-00	A11= 7.1833E-07	A30= 9.3140E+01
					A03= 3.3333E-10	A21= 1.3743E-05	A12= 1.3538E-07
4	3	NC			A1= 1.1510E-11	A2= -2.0700E-13	A3= 4.9345E-15

SOURCE INFORMATION:

FROM 1 TO 0 IMPEDANCE 5.000E+01 R

FREQUENCY	VALUE(HZ)	AMPLITUDE	PHASE(DEG)
1	2.500E+06	1.000E+00	0
2	3.000E+06	1.000E+00	0
3	3.500E+06	1.000E+00	0

AUGMENTED LINEAR NETWORK DESCRIPTION

BRANCH NUMBER	FROM NODE	TO NODE	ELEMENT TYPE	ELEMENT VALUE	CONTROL BRANCH
1	1	2	C	6.600E-09	-0
14	3	5	C	1.014E-09	0
24	4	3	C	1.152E-11	0
13	4	2	C	1.500E-12	0
5	4	6	C	8.100E-09	-0
8	4	0	C	4.000E-12	-0
3	5	0	R	1.000E-01	-0
4	4	0	R	1.980E+03	-0
6	6	0	R	5.000E+01	-0
16	4	3	R	6.350E+05	0
17	1	0	R	5.000E+01	0
7	2	3	R	1.010E+01	0
2	2	0	R	2.020E+04	-0
19	3	5	G	4.332E-01	0
23	4	3	VC	3.842E-01	14
22	4	3	VC	1.871E-08	13
18	1	0	I	0	0
9	5	0	I	0	0
10	6	0	I	0	0
11	3	5	I	0	0
12	4	3	I	0	0
21	4	2	I	0	0

PORT ASSIGNMENTS:

PORT NUMBER	NODE PAIR FROM	TO
1	1	0
2	5	0
3	6	0
4	3	5
5	4	3
6	4	2

OPEN CIRCUIT IMPEDANCE MATRIX

Z(1, 1):

RESIDUE		EIGENVALUE	
-1.83986E+11+J	0	-3.38784E+12+J	0
-7.32280E+11+J	0	-8.71553E+10+J	0
1.17889E+08+J	0	-1.51107E+09+J	0
-6.14623E+08+J	0	-5.30604E+07+J	0
-5.54126E+07+J	0	-1.84473E+06+J	0
6.39202E+00+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(1, 2):

RESIDUE		EIGENVALUE	
-2.49301E+11+J	0	-3.38784E+12+J	0
-6.99546E+08+J	0	-8.71553E+10+J	0
-1.05010E+06+J	0	-1.51107E+09+J	0
1.18303E+06+J	0	-5.30604E+07+J	0
1.10707E+05+J	0	-1.84473E+06+J	0
-1.12321E-02+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(1, 3):

RESIDUE		EIGENVALUE	
-1.84519E+11+J	0	-3.38784E+12+J	0
-6.60680E+10+J	0	-3.71553E+10+J	0
5.62370E+08+J	0	-1.51107E+03+J	0
2.42051E+07+J	0	-5.30604E+07+J	0
-2.91765E+04+J	0	-1.84473E+06+J	0
-9.91836E-01+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(1, 4):

RESIDUE		EIGENVALUE	
7.32682E+08+J	0	-3.38784E+12+J	0
-2.05077E+08+J	0	-8.71553E+10+J	0
1.52734E+08+J	0	-1.51107E+03+J	0
-7.04654E+08+J	0	-5.30604E+07+J	0
2.43254E+07+J	0	-1.84473E+06+J	0
-3.06324E+02+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(1, 5):

RESIDUE		EIGENVALUE	
6.40496E+10+J	0	-3.38784E+12+J	0
-6.51615E+10+J	0	-8.71553E+10+J	0
4.09767E+08+J	0	-1.51107E+09+J	0
7.26560E+08+J	0	-5.30604E+07+J	0
-2.44263E+07+J	0	-1.84473E+05+J	0
3.45590E+02+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(1, 6):

RESIDUE		EIGENVALUE	
-5.32377E+08+J	0	-3.33784E+12+J	0
6.66188E+11+J	0	-3.71553E+10+J	0
4.43799E+08+J	0	-1.51107E+03+J	0
6.02600E+03+J	0	-5.30604E+07+J	0
-3.56028E+07+J	0	-1.84473E+03+J	0
3.46727E+02+J	0	-6.16155E+04+J	0

```
CONSTANT=              0
I(  2,  1):
          RESIDUE                               EIGENVALUE
    -2.49301E+11+J          0    -3.38784E+12+J          0
    -6.92314E+03+J          0    -3.71553E+10+J          0
     1.75427E+07+J          0    -1.51107E+03+J          0
    -2.50471E+07+J          0    -5.30504E+07+J          0
     1.01003E+03+J          0    -1.84472E+13+J          0
    -1.04794E+01+J          0    -6.16155E+04+J          0

CONSTANT=              0
I(  2,  2):
          RESIDUE                               EIGENVALUE
    -3.27808E+11+J          0    -3.38784E+12+J          0
    [illegible]             0    -3.71553E+10+J          [illegible]
    [illegible]             [illegible]    -1.51107E+03+J          0
    [illegible]             [illegible]    [illegible]             [illegible]
    [illegible]             [illegible]    -1.84472E+13+J          [illegible]
    [illegible]             [illegible]    -6.16155E+04+J          [illegible]

[illegible]

[illegible]

CONSTANT=              0
[illegible]
          RESIDUE                               EIGENVALUE
    [illegible]             0    -3.38784E+12+J          0
    [illegible]             0    -3.71553E+10+J          0
    [illegible]             0    -1.51107E+03+J          [illegible]
    [illegible]             0    -5.30504E+07+J          [illegible]
    [illegible]             0    -1.84472E+13+J          [illegible]
    [illegible]             0    -6.16155E+04+J          [illegible]

CONSTANT=              0
I(  2,  6):
          RESIDUE                               EIGENVALUE
    -7.21370E+03+J          0    -3.38784E+12+J          0
     6.88830E+03+J          0    -3.71553E+10+J          0
     6.00407E+07+J          0    -1.51107E+03+J          0
     2.48513E+07+J          0    -5.30504E+07+J          0
     6.49327E+05+J          0    -1.84472E+13+J          0
    -7.01172E+03+J          0    -6.16155E+04+J          0
```

```
CONSTANT=             0
Z(  3,   1):
           RESIDUE                                     EIGENVALUE
      -1.84519E+11+J          0     -3.33784E+12+J            0
      -6.50476E+10+J          0     -8.71553E+10+J            0
      -1.22870E+10+J          0     -1.51107E+09+J            0
       1.31066E+10+J          0     -5.30604E+07+J            0
      -4.53808E+08+J          0     -1.84473E+06+J            0
       4.88821E+05+J          0     -6.16155E+04+J            0

CONSTANT=             0
Z(  3,   2):
           RESIDUE                                     EIGENVALUE
      -2.50022E+11+J          0     -3.33784E+12+J            0
      -6.20041E+07+J          0     -8.71553E+10+J            0
       1.09447E+08+J          0     -1.51107E+09+J            0
      -2.52277E+07+J          0     -5.30604E+07+J            0
       9.06945E+05+J          0     -1.84473E+06+J            0
      -3.58307E+02+J          0     -6.16155E+04+J            0

CONSTANT=             0
Z(  3,   3):
           RESIDUE                                     EIGENVALUE
      -1.85052E+11+J          0     -3.33784E+12+J            0
      -5.94002E+09+J          0     -8.71553E+10+J            0
      -5.83134E+10+J          0     -1.51107E+09+J            0
      -5.16167E+08+J          0     -5.30604E+07+J            0
      -2.38344E+05+J          0     -1.84473E+06+J            0
      -7.58184E+04+J          0     -6.16155E+04+J            0

CONSTANT=             0
Z(  3,   4):
           RESIDUE                                     EIGENVALUE
       7.34802E+08+J          0     -3.33784E+12+J            0
      -1.84408E+07+J          0     -8.71553E+10+J            0
      -1.59189E+10+J          0     -1.51107E+09+J            0
       1.50267E+10+J          0     -5.30604E+07+J            0
       1.99216E+08+J          0     -1.84473E+06+J            0
      -2.34544E+07+J          0     -6.16155E+04+J            0

CONSTANT=             0
Z(  3,   5):
           RESIDUE                                     EIGENVALUE
       6.42349E+10+J          0     -3.33784E+12+J            0
      -5.85941E+09+J          0     -8.71553E+10+J            0
      -4.27083E+10+J          0     -1.51107E+09+J            0
      -1.54937E+10+J          0     -5.30604E+07+J            0
      -2.00041E+08+J          0     -1.84473E+06+J            0
       2.64177E+07+J          0     -6.16155E+04+J            0

CONSTANT=             0
Z(  3,   6):
           RESIDUE                                     EIGENVALUE
      -5.33918E+08+J          0     -3.33784E+12+J            0
       5.99045E+10+J          0     -8.71553E+10+J            0
      -4.62553E+10+J          0     -1.51107E+09+J            0
      -1.28502E+10+J          0     -5.30604E+07+J            0
      -2.91572E+08+J          0     -1.84473E+06+J            0
       2.65047E+07+J          0     -6.16155E+04+J            0
```

```
CONSTANT=              0
Z(  4,  1):
                RESIDUE                              EIGENVALUE
       7.25520E+08+J            0      -3.38784E+12+J              0
       7.89932E+07+J            0      -3.71553E+10+J              0
      -1.55536E+08+J            0      -1.51107E+09+J              0
      -3.62107E+08+J            0      -5.30604E+07+J              0
       2.34293E+07+J            0      -1.84472E+06+J              0
      -3.11195E+02+J            0      -6.16155E+04+J              0

CONSTANT=              0
Z(  4,  2):
                RESIDUE                              EIGENVALUE
       9.83077E+03+J            0      -3.38784E+12+J              0
       7.51755E+04+J            0      -3.71553E+10+J              0
       1.42102E+03+J            0      -1.51107E+03+J              0
       1.28397E+03+J            0      -5.30604E+07+J              0
      -4.68095E+04+J            0      -1.84473E+05+J              0
       5.46333E-01+J            0      -6.16155E+04+J              0

CONSTANT=              0
Z(  4,  [illegible]):
                RESIDUE                              EIGENVALUE
       7.[illegible]+J          0      -3.38784E+12+J              0
       7.[illegible]+J          0      -3.71553E+10+J              0
      -[illegible]+J            0      -1.51107E+[illegible]+J     0
       [illegible]              0      -5.30604E+[illegible]+J     0
       1.[illegible]            0      -1.84473E+06+J              0
       [illegible]              0      -6.16155E+04+J              0

[illegible]                    [illegible]
[illegible]
                [illegible]                           EIGENVALUE
       [illegible]              0      -3.38784E+12+J              [illegible]
       [illegible]              0      -3.71553E+10+J              [illegible]
      -[illegible]              0      -1.51107E+[illegible]+J     [illegible]
      -[illegible]+J            [illegible]  -[illegible]          0
      -1.[illegible]            0      -1.84472E+06+J              0
       [illegible]+04+J         0      -6.16155E+04+J              [illegible]

CONSTANT=              0
Z(  4,  [illegible]):
                RESIDUE                              EIGENVALUE
      -2.98399E+03+J            0      -3.38784E+12+J              [illegible]
       7.0[illegible]+0[illegible]+J  0  -3.71553E+10+J            0
      -[illegible]+03+J         0      -1.51107E+09+J              [illegible]
       [illegible]+03+J         0      -5.30604E+07+J              [illegible]
       1.[illegible]+07+J       0      -1.84473E+09+J              [illegible]
      -1.[illegible]+04+[illegible]  0  -6.161[illegible]+04+J     [illegible]

CONSTANT=              0
Z(  4,  [illegible]):
                RESIDUE                              EIGENVALUE
       2.0[illegible]+03+J      0      -3.38784E+12+J              0
      -7.15908E+07+J            0      -3.71553E+10+J              0
      -6.0[illegible]+03+J      0      -1.51107E+03+J              0
       6.55033E+09+J            0      -5.30604E+07+J              0
       1.[illegible]+07+J       0      -1.84473E+05+J              0
      -1.[illegible]+04+J       0      -6.16155E+04+J              0
```

CONSTANT= 0

Z(5, 1):

RESIDUE		EIGENVALUE	
6.40568E+10+J	0	-3.38784E+12+J	0
-6.52321E+10+J	0	-8.71553E+10+J	0
-1.21250E+10+J	0	-1.51107E+09+J	0
1.31902E+10+J	0	-5.30604E+07+J	0
1.29165E+08+J	0	-1.84473E+06+J	0
-1.90917E+07+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(5, 2):

RESIDUE		EIGENVALUE	
8.67967E+10+J	0	-3.38784E+12+J	0
-6.23161E+07+J	0	-8.71553E+10+J	0
1.08004E+08+J	0	-1.51107E+09+J	0
-2.53885E+07+J	0	-5.30604E+07+J	0
-2.58054E+05+J	0	-1.84473E+06+J	0
3.35480E+04+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(5, 3):

RESIDUE		EIGENVALUE	
6.42421E+10+J	0	-3.38784E+12+J	0
-5.88539E+09+J	0	-8.71553E+10+J	0
-5.78403E+10+J	0	-1.51107E+09+J	0
-5.19457E+08+J	0	-5.30604E+07+J	0
6.80094E+04+J	0	-1.84473E+06+J	0
2.96242E+06+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(5, 4):

RESIDUE		EIGENVALUE	
-2.55091E+08+J	0	-3.38784E+12+J	0
-1.82684E+07+J	0	-3.71553E+10+J	0
-1.57089E+10+J	0	-1.51107E+09+J	0
1.51225E+10+J	0	-5.30604E+07+J	0
-5.67018E+07+J	0	-1.84473E+06+J	0
9.16425E+08+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(5, 5):

RESIDUE		EIGENVALUE	
-2.22995E+10+J	0	-3.38784E+12+J	0
-5.80464E+09+J	0	-8.71553E+10+J	0
-4.21449E+10+J	0	-1.51107E+09+J	0
-1.55924E+10+J	0	-5.30604E+07+J	0
5.69358E+07+J	0	-1.84473E+06+J	0
-1.03221E+09+J	0	-6.16155E+04+J	0

CONSTANT= 0

Z(5, 6):

RESIDUE		EIGENVALUE	
1.85353E+08+J	0	-3.38784E+12+J	0
5.93446E+10+J	0	-3.71553E+10+J	0
-4.55451E+10+J	0	-1.51107E+09+J	0
-1.29322E+10+J	0	-5.30604E+07+J	0
8.29839E+07+J	0	-1.84473E+06+J	0
-1.03551E+09+J	0	-6.16155E+04+J	0

CONSTANT= 0
Z(6, 1):

RESIDUE		EIGENVALUE	
-5.32521E+08+J	0	-3.38784E+12+J	0
6.66409E+11+J	0	-8.71553E+10+J	0
-1.22846E+10+J	0	-1.51107E+09+J	0
1.30763E+10+J	0	-5.30604E+07+J	0
1.17993E+08+J	0	-1.84473E+05+J	0
-1.90917E+07+J	0	-6.16155E+04+J	0

CONSTANT= 0
Z(6, 2):

RESIDUE		EIGENVALUE	
-7.21554E+03+J	0	-3.38784E+12+J	0
6.36619E+08+J	0	-8.71553E+10+J	0
1.10316E+08+J	0	-1.51107E+09+J	0
-2.51692E+07+J	0	-5.30604E+07+J	0
-2.35733E+05+J	0	-1.84473E+05+J	0
3.35431E+04+J	0	-6.16155E+04+J	0

CONSTANT= 0
Z(6, 3):

RESIDUE		EIGENVALUE	
-5.34062E+08+J	0	-3.38784E+12+J	0
6.01249E+10+J	0	-8.71553E+10+J	0
-5.90789E+10+J	0	-1.51107E+09+J	0
-5.14970E+08+J	0	-5.30604E+07+J	0
[illegible].21269E+04+J	0	-1.84473E+05+J	0
2.56243E+05+J	0	-6.16155E+04+J	0

CONSTANT= 0
Z(6, 4):

RESIDUE		EIGENVALUE	
1.12034E+06+J	0	-3.38784E+12+J	0
1.25530E+08+J	0	-8.71553E+10+J	0
-1.60453E+10+J	0	-1.51107E+09+J	0
1.40319E+10+J	0	-5.30604E+07+J	0
-3.17972E+07+J	0	-1.84473E+05+J	0
9.16426E+06+J	0	-6.16155E+04+J	0

CONSTANT= 0
Z(6, 5):

RESIDUE		EIGENVALUE	
1.85382E+08+J	0	-3.38784E+12+J	0
5.93000E+10+J	0	-8.71553E+10+J	0
-4.30474E+10+J	0	-1.51107E+09+J	0
-1.54577E+10+J	0	-5.30604E+07+J	0
5.20120E+07+J	0	-1.84473E+05+J	0
-1.03221E+09+J	0	-6.16155E+04+J	0

CONSTANT= 0
Z(6, 6):

RESIDUE		EIGENVALUE	
-1.54089E+06+J	0	-3.38784E+12+J	0
-6.06262E+11+J	0	-8.71553E+10+J	0
-4.66226E+10+J	0	-1.51107E+09+J	0
-1.28205E+10+J	0	-5.30604E+07+J	0
7.58107E+07+J	0	-1.84473E+05+J	0
-1.03561E+09+J	0	-6.16155E+04+J	0

CONSTANT= 0

FIRST ORDER: FREQUENCY(1)= 2.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.00336E-02	-1.32885E-03	1.01212E-02	-3.98954E+01	1.00336E-02	-1.32885E-03	1.01212E-02	-7.54
3	-4.29645E+00	7.73651E-01	4.36555E+00	1.28008E+01	-4.29645E+00	7.73651E-01	4.36555E+00	169.79

FIRST ORDER: FREQUENCY(2)= 3.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	9.77794E-03	-1.94875E-03	9.97024E-03	-4.00259E+01	9.77794E-03	-1.94875E-03	9.97024E-03	-11.27
3	-4.16119E+00	1.07990E+00	4.29904E+00	1.26674E+01	-4.16119E+00	1.07990E+00	4.29904E+00	165.45

FIRST ORDER: FREQUENCY(3)= 3.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	9.47540E-03	-2.46325E-03	9.79034E-03	-4.01840E+01	9.47540E-03	-2.46325E-03	9.79034E-03	-14.57
3	-4.00289E+00	1.33561E+00	4.21984E+00	1.25059E+01	-4.00289E+00	1.33561E+00	4.21984E+00	161.55

SECOND ORDER: FREQUENCY(1, 1)= 5.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.05574E-02	1.72659E-03	2.06397E-02	-3.37059E+01	1.02837E-02	8.63346E-04	1.03199E-02	4.80
3	-9.64119E+00	-1.10589E+00	9.70441E+00	1.97394E+01	-4.82059E+00	-5.52945E-01	4.85220E+00	-173.46

SECOND ORDER: FREQUENCY(1, 2)= 5.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.10787E-02	1.72983E-04	2.10794E-02	-3.35228E+01	2.10787E-02	1.72983E-04	2.10794E-02	.47
3	-9.93577E+00	-3.65359E-01	9.94248E+00	1.99499E+01	-9.93577E+00	-3.65359E-01	9.94243E+00	-177.89

SECOND ORDER: FREQUENCY(1, 3)= 6.000E+06 HZ

TRANSFER FUNCTION OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.13517E-02	-1.32933E-03	2.13931E-02	-3.33945E+01	2.13517E-02	-1.32965E-03	2.13931E-02	-3.59
3	-1.01112E+01	3.60322E-01	1.01176E+01	2.01015E+01	-1.01112E+01	3.60322E-01	1.01176E+01	177.99

SECOND ORDER: FREQUENCY(1,-1)= 0 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	3.05822E-03	0	3.05822E-03	-5.02905E+01	3.05822E-03	0	3.05822E-03	0
3	6.60906E-11	0	6.60906E-11	-2.03597E+02	6.60906E-11	0	6.60906E-11	0

SECOND ORDER: FREQUENCY(2, 2)= 5.000E+05 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.14103E-02	-1.43291E-03	2.14327E-02	-3.33563E+01	1.07053E-02	-7.43252E-04	1.07312E-02	-3.83
3	-1.01411E+01	4.37652E-01	1.01505E+01	2.01297E+01	-5.07053E+00	2.18826E-01	5.07524E+00	177.53

SECOND ORDER: FREQUENCY(2, 3)= 6.500E+05 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.14893E-02	-3.09934E-03	2.17178E-02	-3.32637E+01	2.14893E-02	-3.09934E-03	2.17178E-02	-3.13
3	-1.02240E+01	1.20434E+00	1.02947E+01	2.02523E+01	-1.02240E+01	1.20434E+00	1.02947E+01	173.28

SECOND ORDER: FREQUENCY(2,-1)= 5.000E+05 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.14977E-02	4.92549E-03	1.25083E-02	-3.80560E+01	1.14977E-02	4.92549E-03	1.25083E-02	23.19
3	-5.11863E+00	-2.30334E+00	5.61300E+00	1.49839E+01	-5.11863E+00	-2.30334E+00	5.61300E+00	-155.77

SECOND ORDER: FREQUENCY(2,-2)= 0 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG

2	2.96590E-03	0	2.96590E-03	-5.05569E+01	2.96590E-03	0	2.96590E-03	0
3	6.40957E-11	0	6.40957E-11	-2.03863E+02	6.40957E-11	0	6.40957E-11	0

SECOND ORDER: FREQUENCY(3, 2)= 7.000E+06 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.14091E-02	-4.68217E-03	2.19151E-02	-3.31851E+01	1.07045E-02	-2.34109E-03	1.09575E-02	-12.34
3	-1.02152E+01	1.99344E+00	1.04083E+01	2.03478E+01	-5.10791E+00	9.96720E-01	5.20424E+00	168.96

SECOND ORDER: FREQUENCY(3,-1)= 1.000E+05 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.25117E-02	3.06514E-03	1.28532E-02	-3.71377E+01	1.25117E-02	3.06514E-03	1.28532E-02	12.79
3	-6.04923E+00	-1.51459E+00	6.23593E+00	1.58981E+01	-6.04923E+00	-1.51459E+00	6.23593E+00	-165.94

SECOND ORDER: FREQUENCY(3,-2)= 5.000E+05 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.10789E-02	4.84409E-03	1.20916E-02	-3.33503E+01	1.10789E-02	4.84409E-03	1.20916E-02	23.62
3	-4.93136E+00	-2.26341E+00	5.42593E+00	1.46893E+01	-4.93136E+00	-2.26341E+00	5.42593E+00	-155.35

SECOND ORDER: FREQUENCY(3,-3)= 0 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.85703E-03	0	2.85703E-03	-5.08793E+01	2.85703E-03	0	2.85703E-03	0
3	6.17608E-11	0	6.17608E-11	-2.04186E+02	6.17608E-11	0	6.17608E-11	0

THIRD ORDER: FREQUENCY(1, 1, 1)= 7.500E+03 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
[illegible]	4.22191E-02	-5.22017E-02	6.72... E-02	-2.34... E+01	1.05703E-02	-1.30720E-02	1.68117E-02	-51.04
[illegible]	[illegible]	2.43... E+01	3.1557... E+01	3.00... E+01	-5.16... E+00	6.05756E+00	7.9... E+00	130.25

THIRD ORDER: FREQUENCY(1, 1, 2)= 3.000E+05 HZ

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
	TRANSFER FUNCTION				OUTPUT VOLTAGE			
2	3.53413E-02	-5.67217E-02	6.68308E-02	-2.35005E+01	2.65060E-02	-4.25413E-02	5.01231E-02	-58.07
3	-1.73509E+01	2.66435E+01	3.18055E+01	3.00500E+01	-1.30274E+01	1.99826E+01	2.38541E+01	123.10

THIRD ORDER: FREQUENCY(1, 1, 3)= 8.500E+09 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.83151E-02	-5.95128E-02	6.59057E-02	-2.36097E+01	2.12364E-02	-4.47096E-02	4.94968E-02	-64.59
3	-1.40192E+01	2.81509E+01	3.14483E+01	2.99520E+01	-1.05144E+01	2.11132E+01	2.35864E+01	116.47

THIRD ORDER: FREQUENCY(1, 1,-1)= 2.500E+09 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.[illegible]256E-02	5.68922E-03	6.28810E-02	-2.32405E+01	5.14592E-02	4.26691E-03	5.16458E-02	4.74
3	-3.13334E+01	-3.70764E+00	3.15268E+01	2.99736E+01	-2.34738E+01	-2.84073E+00	2.36451E+01	-173.10

THIRD ORDER: FREQUENCY(1, 1,-2)= 2.000E+09 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.38235E-02	1.57547E-02	4.65694E-02	-2.66380E+01	3.28676E-02	1.18160E-02	3.49271E-02	19.77
3	-1.96124E+01	-7.83166E+00	2.11363E+01	2.65008E+01	-1.47093E+01	-5.91125E+00	1.58527E+01	-153.11

THIRD ORDER: FREQUENCY(1, 1,-3)= 1.500E+09 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	3.64431E-02	1.44068E-02	3.91874E-02	-2.81371E+01	2.73323E-02	1.08051E-02	2.93906E-02	21.57
3	-1.61755E+01	-7.02880E+00	1.76367E+01	2.49284E+01	-1.21317E+01	-5.27160E+00	1.32276E+01	-156.51

THIRD ORDER: FREQUENCY(1, 2, 2)= 8.500E+09 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.79005E-02	-6.01266E-02	6.62846E-02	-2.35718E+01	2.09254E-02	-4.50949E-02	4.97134E-02	-65.11
3	-1.38261E+01	2.83994E+01	3.15861E+01	2.99899E+01	-1.03696E+01	2.12995E+01	2.36896E+01	115.96

THIRD ORDER: FREQUENCY(1, 2, 3)= 9.000E+06 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.06105E-02	-6.20096E-02	6.53452E-02	-2.36957E+01	3.09158E-02	-9.30145E-02	9.80177E-02	-71.61
3	-1.03270E+01	2.94131E+01	3.11733E+01	2.98757E+01	-1.54905E+01	4.41196E+01	4.67593E+01	109.35

THIRD ORDER: FREQUENCY(1, 2,-1)= 3.000E+05 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.21546E-02	-7.10093E-03	6.25589E-02	-2.40742E+01	9.32319E-02	-1.06514E-02	9.38334E-02	-6.52
3	-2.87286E+01	2.16909E+00	2.88104E+01	2.91910E+01	-4.30929E+01	3.25363E+00	4.32156E+01	175.68

THIRD ORDER: FREQUENCY(1, 2,-2)= 2.500E+05 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	5.56395E-02	9.13512E-03	5.63844E-02	-2.49768E+01	8.34592E-02	1.37027E-02	8.45765E-02	9.32
3	-2.52819E+01	-5.15243E+00	2.58016E+01	2.82329E+01	-3.79228E+01	-7.72873E+00	3.87023E+01	-168.43

THIRD ORDER: FREQUENCY(1, 2,-3)= 2.000E+05 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	3.90422E-02	1.32317E-02	4.13883E-02	-2.75374E+01	5.97634E-02	1.98776E-02	6.29824E-02	18.40
3	[illegible]	[illegible]	1.90503E+01	2.55983E+01	-2.67603E+01	-1.00243E+01	2.85762E+01	-159.45

THIRD ORDER: FREQUENCY(1, 3, 3)= 9.500E+05 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	[illegible]	[illegible]	9.43273E-02	-2.38320E+01	9.94693E-03	-4.72092E-02	4.82457E-02	-78.10
3	[illegible]	[illegible]	3.07177E+01	2.97478E+01	-5.08739E+00	2.24695E+01	2.30383E+01	102.76

THIRD ORDER: FREQUENCY(1, 3,-1)= 3.500E+05 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.02484E-02	-1.05795E-02	6.11876E-02	-2.42667E+01	9.03726E-02	-1.60192E-02	9.17814E-02	-10.05
3	-2.80893E+01	3.87244E+00	2.83552E+01	2.90530E+01	-4.21346E+01	5.81766E+00	4.25344E+01	172.14

THIRD ORDER: FREQUENCY(1, 3,-2)= 3.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.90593E-02	-2.03728E-03	4.91040E-02	-2.61777E+01	7.35895E-02	-3.13106E-03	7.36559E-02	-2.44
3	-2.25337E+01	7.90857E-02	2.25338E+01	2.70798E+01	-3.38905E+01	1.18628E-01	3.38907E+01	179.80

THIRD ORDER: FREQUENCY(1, 3,-3)= 2.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	5.08258E-02	7.29230E-03	5.13474E-02	-2.57896E+01	7.62388E-02	1.09497E-02	7.70211E-02	8.17
3	-2.3[illegible]E+01	-4.23262E+00	2.34315E+01	2.74182E+01	-3.46606E+01	-6.34893E+00	3.52372E+01	-169.62

THIRD ORDER: FREQUENCY(2, 2, 2)= 9.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	[illegible]	[illegible]	5.06136E-02	-2.58594E+01	5.03384E-02	-1.56134E-02	1.64043E-02	-72.12
3	[illegible]	[illegible]	3.13033E+01	2.99120E+01	-2.52599E+00	7.40712E+00	7.82533E+00	108.33

THIRD ORDER: FREQUENCY(2, 2, 3)= 9.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	[illegible]	-3.23158E-02	3.45063E-02	-2.37972E+01	9.56046E-03	-4.74869E-02	4.84397E-02	-78.62
3	-3.53372E+00	3.61393E+01	3.62111E+01	2.97826E+01	-4.[illegible]E+00	2.28049E+01	2.31308E+01	102.24

THIRD ORDER: FREQUENCY(2, 2,-1)= 3.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	5.37348E-02	-1.98713E-02	5.72914E-02	-2.48382E+01	4.03011E-02	-1.49035E-02	4.29685E-02	-20.29
3	-2.51982E+01	8.25516E+00	2.65160E+01	2.84702E+01	-1.88987E+01	6.19137E+00	1.98870E+01	161.86

THIRD ORDER: FREQUENCY(2, 2,-2)= 3.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.75850E-02	9.47438E-04	6.75917E-02	-2.34021E+01	5.06888E-02	7.10579E-04	5.06937E-02	.80
3	-3.11214E+01	-1.63671E+00	3.11644E+01	2.98732E+01	-2.33411E+01	-1.22753E+00	2.33733E+01	-176.99

THIRD ORDER: FREQUENCY(2, 2,-3)= 2.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.33241E-02	1.20875E-02	4.49787E-02	-2.69399E+01	3.24930E-02	9.06564E-03	3.37340E-02	15.59
3	-1.95821E+01	-6.30260E+00	2.05713E+01	2.62653E+01	-1.46866E+01	-4.72695E+00	1.54285E+01	-162.16

THIRD ORDER: FREQUENCY(2, 3, 3)= 1.000E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	5.44814E-03	-6.32511E-02	6.34853E-02	-2.39465E+01	4.08610E-03	-4.74383E-02	4.76140E-02	-85.08
3	-3.00544E+00	3.01919E+01	3.03411E+01	2.96406E+01	-2.25408E+00	2.26439E+01	2.27553E+01	95.68

THIRD ORDER: FREQUENCY(2, 3,-1)= 4.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	5.05720E-02	-2.32806E-02	5.56733E-02	-2.50871E+01	7.58580E-02	-3.49209E-02	8.35099E-02	-24.72
3	-2.39143E+01	9.96256E+00	2.59065E+01	2.82682E+01	-3.58715E+01	1.49438E+01	3.88597E+01	157.33

THIRD ORDER: FREQUENCY(2, 3,-2)= 3.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.00699E-02	-1.10751E-02	6.10823E-02	-2.42817E+01	9.01048E-02	-1.66126E-02	9.16234E-02	-10.45
3	-2.80146E+01	4.06848E+00	2.83085E+01	2.90383E+01	-4.20219E+01	6.10272E+00	4.24627E+01	171.74

THIRD ORDER: FREQUENCY(2, 3,-3)= 3.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	5.44314E-02	5.25508E-03	5.47145E-02	-2.52380E+01	8.16922E-02	7.83413E-03	8.20717E-02	5.51
3	-2.49353E+01	-3.40132E+00	2.52158E+01	2.80335E+01	-3.74780E+01	-5.10199E+00	3.78237E+01	-172.25

THIRD ORDER: FREQUENCY(3, 3, 3)= 1.050E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.63324E-03	-6.22039E-02	6.23251E-02	-2.41067E+01	-4.09059E-04	-1.55759E-02	1.55813E-02	-91.50
3	4.35063E-01	2.93036E+01	2.93082E+01	2.94869E+01	1.08766E-01	7.45140E+00	7.45219E+00	89.16

THIRD ORDER: FREQUENCY(3, 3,-1)= 4.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.68335E-02	-2.64545E-02	5.37929E-02	-2.53855E+01	3.51288E-02	-1.98409E-02	4.03447E-02	-29.46
3	-2.30165E+01	1.15378E+01	2.51474E+01	2.80093E+01	-1.67388E+01	8.69086E+00	1.88605E+01	152.56

THIRD ORDER: FREQUENCY(3, 3,-2)= 4.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	5.07245E-02	-2.28985E-02	5.56533E-02	-2.50901E+01	3.80434E-02	-1.71746E-02	4.17404E-02	-24.30
3	-2.39301E+01	9.78644E+00	2.59002E+01	2.82661E+01	-1.79851E+01	7.33983E+00	1.94251E+01	157.80

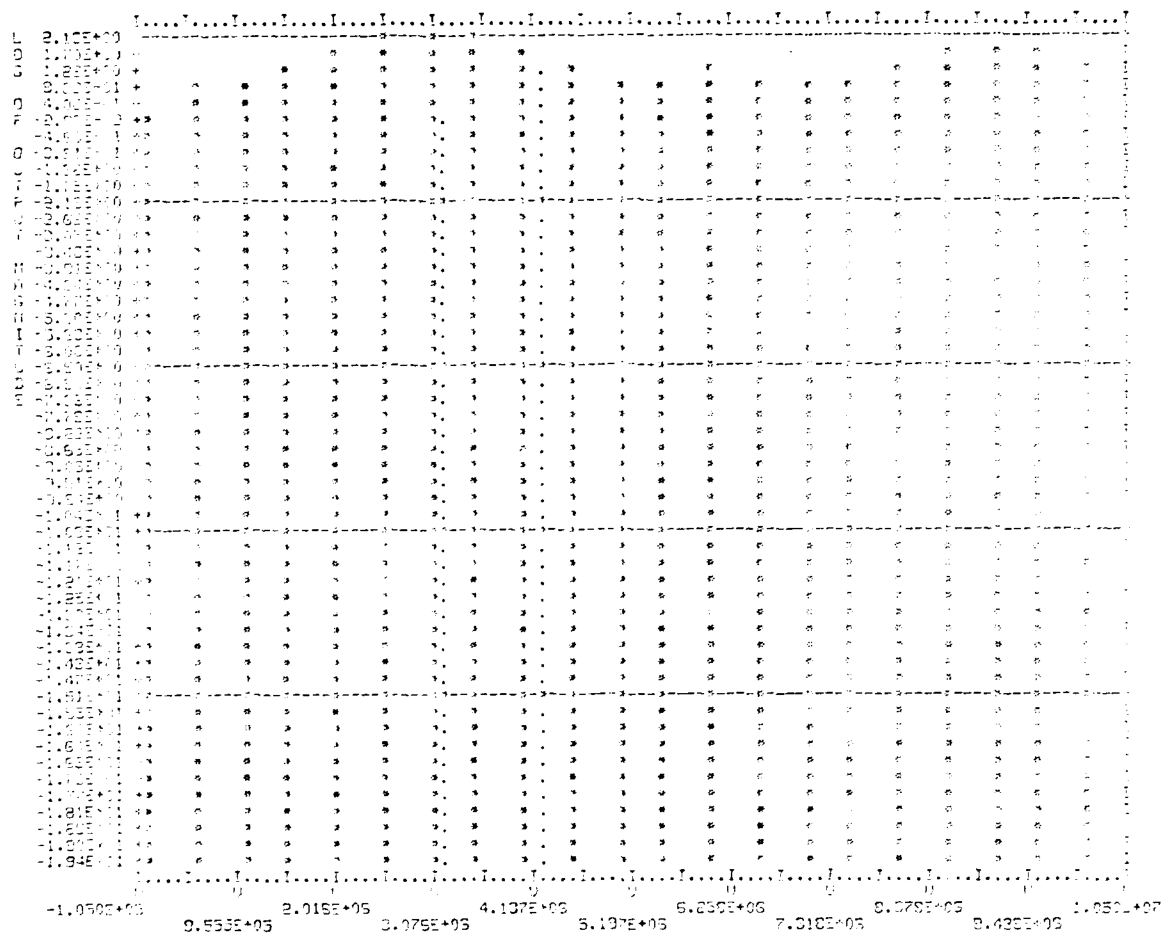
THIRD ORDER: FREQUENCY(3, 3,-3)= 3.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.55980E-02	-3.13490E-03	6.56722E-02	-2.36523E+01	4.91985E-02	-2.35095E-03	4.92546E-02	-2.74
3	-3.04699E+01	2.92175E-01	3.04713E+01	2.96778E+01	-2.28524E+01	2.19131E-01	2.28535E+01	179.45

SINUSOIDAL STEADY-STATE OUTPUT RESPONSE AT PORT 3

FREQUENCY HERTZ	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2.50E+06	-1.064E+02	-2.242E+01	1.088E+02	-1.681E+02
3.00E+06	-1.336E+02	-4.037E+00	1.337E+02	-1.783E+02
3.50E+06	-1.219E+02	1.700E+01	1.231E+02	1.721E+02
5.00E+06	-4.821E+00	-5.529E-01	4.852E+00	-1.735E+02
5.50E+06	-9.936E+00	-3.654E-01	9.942E+00	-1.779E+02
6.00E+06	-1.518E+01	5.789E-01	1.519E+01	1.778E+02
0	1.919E-10	0	1.919E-10	0
6.50E+06	-1.022E+01	1.204E+00	1.029E+01	1.733E+02
5.00E+05	-1.005E+01	-4.567E+00	1.104E+01	-1.556E+02
7.00E+06	-5.108E+00	9.967E-01	5.204E+00	1.690E+02
1.00E+06	-6.049E+00	-1.515E+00	6.236E+00	-1.659E+02
7.50E+06	-5.162E+00	6.098E+00	7.989E+00	1.302E+02
8.00E+06	-1.303E+01	1.998E+01	2.385E+01	1.231E+02
8.50E+06	-2.088E+01	4.241E+01	4.728E+01	1.162E+02
2.00E+06	-4.147E+01	-1.594E+01	4.443E+01	-1.590E+02
1.50E+06	-1.213E+01	-5.272E+00	1.323E+01	-1.565E+02
9.00E+06	-1.802E+01	5.153E+01	5.459E+01	1.093E+02
9.50E+06	-9.992E+00	4.507E+01	4.617E+01	1.025E+02
1.00E+07	-2.254E+00	2.264E+01	2.276E+01	9.568E+01
4.00E+06	-5.386E+01	2.228E+01	5.828E+01	1.575E+02
1.05E+07	1.088E-01	7.451E+00	7.452E+00	8.916E+01
4.50E+06	-1.674E+01	8.691E+00	1.886E+01	1.526E+02

RESPONSE MAGNITUDE VS FREQUENCY



FREQUENCY (HERTZ)

```
TIME FOR FORMING ZOC(SEC)     1.1530
TIME FOR OBTAINING OUTPUT SPECTRUM(SEC)     3.6050
TOTAL EXECUTION TIME(SEC)     4.7580

                                        *** P R A N C ***
                                     SEPTEMBER 1979 VERSION
```

Example 4-2: Two-Stage Tuned Amplifier Circuit

Consider the two-stage tuned amplifier circuit of Fig. 4-8. The input source comprises of two frequencies:

$$v_s(t) = \cos(2\pi\ 3 \times 10^6 t) + \cos(2\pi\ 3.25 \times 10^6 t)$$

The sequence of data cards used are shown in Fig. 4-9. In this example the frequency sweep capability (FS on the option card) offered by PRANC was used.

The computer printed output is similar to that for Example 4-1. The two transistors in the circuit account for six nonlinear elements. Altogether nine ports were extracted for the Volterra series analysis, two of which were the desired output ports.

The maximum number of frequency increments specified were five. Note that, as the frequency sweep is implemented, the set of input frequencies are printed before the transfer function and output voltage values. Considering the execution times, we note that the formation of the 9x9 open-circuit matrix took approximately 4 seconds on the CDC 6500 computer; the calculation and the printing of the transfer functions and output voltage values at the positive frequency values (approximately 90 points*) required approximately 18 seconds. The entire program execution required less than 22 seconds.

*The actual number of points is approximately 150, since transfer functions at negative frequencies are required in the calculations.

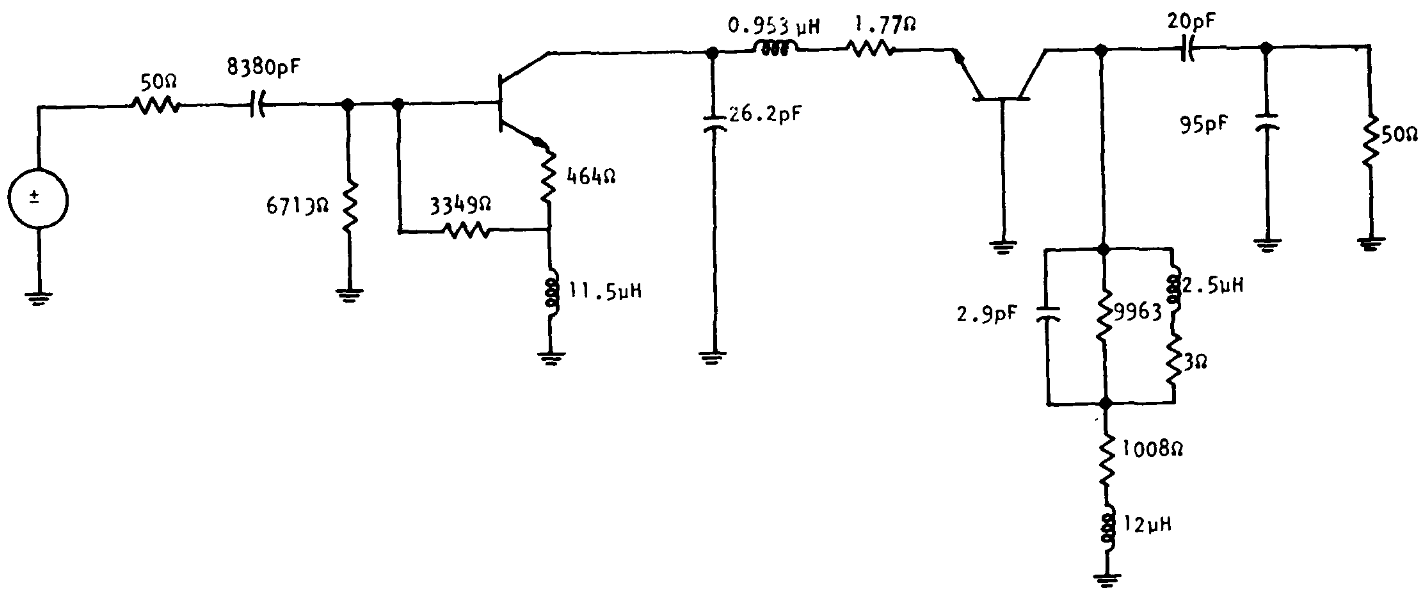


Figure 4-8. Circuit Diagram for Example 4-2

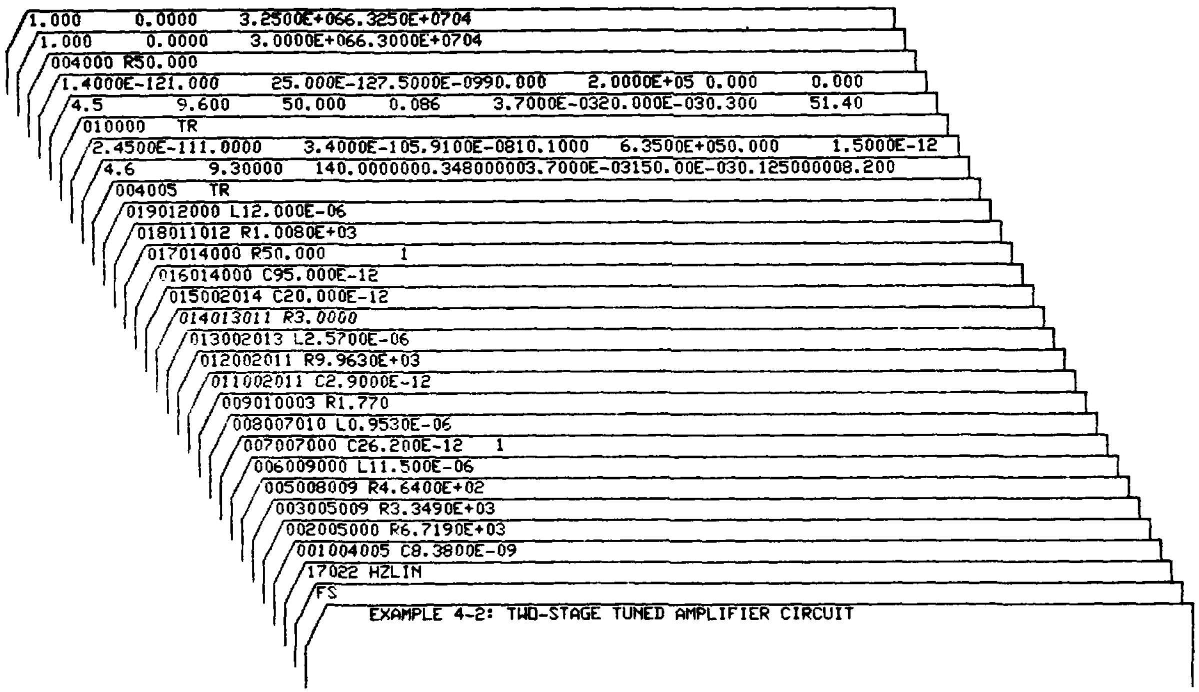


Figure 4-9. Data Cards for Example 4-2

EXAMPLE 4-2: TWO-STAGE TUNED AMPLIFIER CIRCUIT

USER REQUESTED OPTIONS:
DEBUG PRINT-OUT: NO
FREQUENCY SWEEP CAPABILITY: YES
TWO-INPUT CIRCUIT: NO
STATE EQUATION PRINT-OUT: NO
EIGENVALUES MODAL MATRIX PRINT-OUT: NO
OPEN-CIRCUIT IMPEDANCE MATRIX PRINT-OUT: NO
COMPLETE OUTPUT SPECTRUM PLOT: NO
ALL EXTRACTED PORT OUTPUTS: NO

NETWORK DESCRIPTION:
....

LINEAR ELEMENTS

BRANCH NUMBER	FROM NODE	TO NODE	ELEMENT TYPE	ELEMENT VALUE	CONTROL BRANCH
1	4	5	C	8.330E-09	
2	5	0	R	6.715E+03	
3	5	9	R	3.349E+03	
5	8	9	R	4.640E+02	
6	9	0	L	1.150E-05	
7	7	0	C	2.620E-11	
8	7	10	L	9.530E-07	
9	10	3	R	1.770E+00	
11	2	11	C	2.900E-12	
12	2	11	R	9.963E+03	
13	2	13	L	2.570E-06	
14	13	11	R	3.000E+00	
15	2	14	C	2.000E-11	
16	14	0	C	9.500E-11	
18	11	12	R	1.008E+03	
17	12	0	L	1.200E-05	
19	14	0	R	5.000E+01	

TRANSISTOR PARAMETERS:

N= 4.600	VCB= 9.300	VCBO=140.000	MU= .348
IC= 3.700E-03	ICMAX= 1.500E-01	A= .125	HFEMAX= 8.20
K= 2.450E-11	REF= 1.00	CJE= 3.400E-10	C≠2= 5.910E-08
RB= 10.100	RC= 6.350E+05	C1= 0	C3= 1.500E-12

TRANSISTOR PARAMETERS:

N= 4.500	VCB= 9.600	VCBO= 50.000	MU= .086
IC= 3.700E-03	ICMAX= 2.000E-02	A= .300	HFEMAX= 51.40
K= 1.400E-12	REF= 1.00	CJE= 2.500E-11	C≠2= 7.500E-09
RB= 90.000	RC= 2.000E+05	C1= 0	C3= 0

NONLINEAR ELEMENTS

FROM NODE	TO NODE	TYPE	CONTROL (1)	CONTROL (2)	POLYNOMIAL COEFFICIENTS		
....	....	....	......	......	....		
6	8	NR			A1= 1.6286E-01	A2= 3.0862E+00	A3= 3.8988E+01
7	6	ND	25	24	A10= 1.4285E-01 A02= 1.3555E-09 A03= 1.2632E-10	A01= 7.0034E-09 A11= 2.7039E-07 A21= 5.1950E-06	A20= 2.7446E+00 A30= 3.4980E+01 A12= 5.2334E-08
7	6	NC			A1= 1.1275E-11	A2= -2.0207E-13	A3= 4.8284E-15
1	3	NR			A1= 1.4340E-01	A2= 2.7174E+00	A3= 3.4330E+01
2	1	ND	31	30	A10= 1.4083E-01 A02= 1.8866E-07 A03= 1.6457E-08	A01= 1.0334E-06 A11= 3.9331E-05 A21= 7.4643E-04	A20= 2.6726E+00 A30= 3.3741E+01 A12= 7.1807E-06
2	1	NC			A1= 1.1525E-12	A2= -2.0009E-14	A3= 4.6318E-16

SOURCE INFORMATION:

FROM 4 TO 0 IMPEDANCE 5.000E+01 R

FREQUENCY	VALUE(HZ)	AMPLITUDE	PHASE(DEG)
....	....	....	....
1	3.000E+06	1.000E+00	0
2	3.250E+06	1.000E+00	0

FREQUENCY SWEEP TYPE: LIN

MAXIMUM NUMBER OF INCREMENTS= 4

AUGMENTED LINEAR NETWORK DESCRIPTION

BRANCH NUMBER	FROM NODE	TO NODE	ELEMENT TYPE	ELEMENT VALUE	CONTROL BRANCH
......	....	....	....	....	....
1	4	5	C	8.380E-09	-0
25	6	8	C	5.940E-10	0
47	2	1	C	1.153E-12	0
15	2	14	C	2.000E-11	-0
16	14	0	C	9.500E-11	-0
11	2	11	C	2.900E-12	-0
7	7	0	C	2.620E-11	-0
31	1	3	C	5.338E-11	0
24	7	5	C	1.500E-12	0
41	7	6	C	1.128E-11	0
9	10	3	R	1.770E+00	-0
10	0	1	R	9.000E+01	0
5	8	9	R	4.640E+02	-0
12	2	11	R	9.963E+03	-0

4	5	6	R	1.010E+01	0
3	5	9	R	3.349E+03	-0
33	2	1	R	2.000E+05	0
2	5	0	R	6.719E+03	-0
19	14	0	R	5.000E+01	-0
42	1	3	G	1.434E-01	0
18	11	12	R	1.008E+03	-0
27	7	6	R	6.350E+05	0
34	4	0	R	5.000E+01	0
36	6	8	G	1.629E-01	0
14	13	11	R	3.000E+00	-0
30	2	0	R	1.000E+06	0
17	12	0	L	1.200E-05	-0
6	9	0	L	1.150E-05	-0
8	7	10	L	9.530E-07	-0
13	2	13	L	2.570E-06	-0
40	7	6	VC	1.428E-01	25
39	7	6	VC	7.003E-09	24
45	2	1	VC	1.033E-06	30
46	2	1	VC	1.408E-01	31
35	4	0	I	0	0
21	14	0	I	0	0
20	7	0	I	0	0
22	6	8	I	0	0
23	7	6	I	0	0
38	7	5	I	0	0
28	1	3	I	0	0
29	2	1	I	0	0
44	2	0	I	0	0

PORT ASSIGNMENTS:

PORT NUMBER	NODE FROM	PAIR TO
1	4	0
2	14	0
3	7	0
4	6	8
5	7	6
6	7	5
7	1	3
8	2	1
9	2	0

FIRST ORDER: FREQUENCY(1)= 3.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.77042E-02	-1.97449E-02	3.40204E-02	-2.93652E+01	-2.77042E-02	-1.97449E-02	3.40204E-02	-144.52
3	-3.77988E-02	-1.54311E-02	4.08273E-02	-2.77810E+01	-3.77988E-02	-1.54311E-02	4.08273E-02	-157.79

FIRST ORDER: FREQUENCY(2)= 3.250E+06 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-3.16702E-02	-1.91258E-02	3.69972E-02	-2.86366E+01	-3.16702E-02	-1.91258E-02	3.69972E-02	-148.87
3	-4.04870E-02	-1.56788E-02	4.34168E-02	-2.72468E+01	-4.04870E-02	-1.56788E-02	4.34168E-02	-158.83

SECOND ORDER: FREQUENCY(1, 1)= 6.000E+06 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.54432E-03	-1.73550E-03	2.32312E-03	-5.26785E+01	-7.72160E-04	-8.67748E-04	1.16155E-03	-131.66
3	-8.01721E-05	-4.81972E-03	4.82039E-03	-4.63384E+01	-4.00861E-05	-2.40986E-03	2.41020E-03	-90.95

SECOND ORDER: FREQUENCY(1, 2)= 6.250E+06 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.80860E-03	-1.72220E-03	2.49740E-03	-5.20502E+01	-1.80860E-03	-1.72220E-03	2.49740E-03	-136.40
3	-3.54335E-04	-4.95729E-03	4.96994E-03	-4.60730E+01	-3.54335E-04	-4.95729E-03	4.96994E-03	-94.09

SECOND ORDER: FREQUENCY(1,-1)= 0 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.23976E-13	0	2.23976E-13	-2.52996E+02	-2.23976E-13	0	2.23976E-13	180.00
3	2.63862E-03	0	2.63862E-03	-5.15725E+01	2.63862E-03	0	2.63862E-03	0

SECOND ORDER: FREQUENCY(2, 2)= 6.500E+06 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.08395E-03	-1.67593E-03	2.67425E-03	-5.14560E+01	-1.04198E-03	-8.37965E-04	1.33712E-03	-141.19
3	-6.43542E-04	-5.08384E-03	5.12441E-03	-4.58071E+01	-3.21771E-04	-2.54192E-03	2.56220E-03	-97.21

SECOND ORDER: FREQUENCY(2,-1)= 2.500E+05 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.02843E-06	-1.54661E-05	1.55002E-05	-9.61932E+01	1.02843E-06	-1.54661E-05	1.55002E-05	-86.20
3	2.73984E-03	-1.55166E-04	2.74423E-03	-5.12316E+01	2.73984E-03	-1.55166E-04	2.74423E-03	-3.24

SECOND ORDER: FREQUENCY(2,-2)= 0 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.15878E-13	0	2.15878E-13	-2.53316E+02	-2.15878E-13	0	2.15878E-13	180.00
3	2.63205E-03	0	2.63205E-03	-5.15941E+01	2.63205E-03	0	2.63205E-03	0

THIRD ORDER: FREQUENCY(1, 1, 1)= 9.000E+06 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.39034E-03	3.61338E-04	1.43652E-03	-5.68537E+01	-3.47584E-04	9.03344E-05	3.59131E-04	165.43
3	-1.51580E-03	-1.62911E-03	2.22523E-03	-5.30525E+01	-3.78950E-04	-4.07277E-04	5.56307E-04	-132.94

THIRD ORDER: FREQUENCY(1, 1, 2)= 9.250E+06 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.38397E-03	4.96255E-04	1.47496E-03	-5.66244E+01	-1.04173E-03	3.72191E-04	1.10622E-03	160.34
3	-1.63825E-03	-1.58553E-03	2.27986E-03	-5.28418E+01	-1.22869E-03	-1.18915E-03	1.70990E-03	-135.94

THIRD ORDER: FREQUENCY(1, 1,-1)= 3.000E+06 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.91686E-05	-1.49523E-04	1.57400E-04	-7.60599E+01	3.68765E-05	-1.12142E-04	1.18050E-04	-71.80
3	6.51691E-04	-6.78205E-04	9.40565E-04	-6.05322E+01	4.88768E-04	-5.08654E-04	7.05424E-04	-46.14

THIRD ORDER: FREQUENCY(1, 1,-2)= 2.750E+06 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.83228E-05	-1.24630E-04	1.33670E-04	-7.74793E+01	3.62421E-05	-9.34723E-05	1.00253E-04	-63.81
3	6.86804E-04	-6.37739E-04	9.37236E-04	-6.05630E+01	5.15103E-04	-4.78304E-04	7.02927E-04	-42.88

THIRD ORDER: FREQUENCY(1, 2, 2)= 9.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.37264E-03	6.31730E-04	1.51103E-03	-5.64145E+01	-1.02948E-03	4.73797E-04	1.13327E-03	155.29
3	-1.76050E-03	-1.53393E-03	2.33501E-03	-5.26342E+01	-1.32037E-03	-1.15045E-03	1.75126E-03	-133.93

THIRD ORDER: FREQUENCY(1, 2,-1)= 3.250E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.59982E-05	-1.78787E-04	1.84861E-04	-7.46631E+01	7.04972E-05	-2.68181E-04	2.77292E-04	-75.27
3	6.35931E-04	-7.37985E-04	9.74215E-04	-6.02269E+01	9.53972E-04	-1.10698E-03	1.45132E-03	-49.25

THIRD ORDER: FREQUENCY(1, 2,-2)= 3.000E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.82729E-05	-1.46950E-04	1.54675E-04	-7.62116E+01	7.24094E-05	-2.20424E-04	2.32013E-04	-71.81
3	6.56543E-04	-6.79441E-04	9.44822E-04	-6.04930E+01	9.84815E-04	-1.01916E-03	1.41723E-03	-45.93

THIRD ORDER: FREQUENCY(2, 2, 2)= 9.750E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.34157E-03	7.65901E-04	1.54480E-03	-5.62225E+01	-3.35392E-04	1.91475E-04	3.85200E-04	150.28
3	-1.88210E-03	-1.47422E-03	2.39074E-03	-5.24294E+01	-4.70526E-04	-3.68555E-04	5.97635E-04	-141.93

THIRD ORDER: FREQUENCY(2, 2,-1)= 3.500E+06 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.11550E-05	-2.10797E-04	2.14777E-04	-7.33602E+01	3.08662E-05	-1.58098E-04	1.61083E-04	-78.95
3	6.16960E-04	-7.99287E-04	1.00970E-03	-5.99161E+01	4.62720E-04	-5.99465E-04	7.57277E-04	-52.34

THIRD ORDER: FREQUENCY(2, 2,-2)= 3.250E+06 HZ

TRANSFER FUNCTION

OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.55150E-05	-1.70850E-04	1.76809E-04	-7.50499E+01	3.41363E-05	-1.28138E-04	1.32607E-04	-75.08
3	6.24579E-04	-7.20036E-04	9.53179E-04	-6.04165E+01	4.68434E-04	-5.40027E-04	7.14885E-04	-49.06

INPUT FREQUENCIES:

FREQUENCY	VALUE(HZ)
1	2.300E+07
2	2.325E+07

FIRST ORDER: FREQUENCY(1)= 2.300E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	3.71975E-01	1.20091E-01	3.90830E-01	-8.15913E+00	3.71975E-01	1.20091E-01	3.90830E-01	17.89
3	-9.00568E-01	9.28940E-01	1.29381E+00	2.23743E+00	-9.00568E-01	9.28940E-01	1.29381E+00	134.11

FIRST ORDER: FREQUENCY(2)= 2.325E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	4.00636E-01	1.04993E-01	4.14159E-01	-7.65652E+00	4.00636E-01	1.04993E-01	4.14159E-01	14.69
3	-9.22164E-01	1.04472E+00	1.39348E+00	2.88209E+00	-9.22164E-01	1.04472E+00	1.39348E+00	131.43

SECOND ORDER: FREQUENCY(1, 1)= 4.600E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.01767E-03	-2.59958E-02	2.60157E-02	-3.16933E+01	-5.08835E-04	-1.29979E-02	1.30078E-02	-92.24
3	6.29283E-02	-5.81745E-02	8.56931E-02	-2.13405E+01	3.14645E-02	-2.90873E-02	4.28465E-02	-42.75

SECOND ORDER: FREQUENCY(1, 2)= 4.625E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.20123E-03	-2.77663E-02	2.78534E-02	-3.11024E+01	-2.20123E-03	-2.77663E-02	2.78534E-02	-94.53
3	6.31337E-02	-6.35062E-02	8.95483E-02	-2.09588E+01	6.31337E-02	-6.35062E-02	8.95483E-02	-45.17

SECOND ORDER: FREQUENCY(1,-1)= 0 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.11713E-14	0	1.11713E-14	-2.79038E+02	-1.11713E-14	0	1.11713E-14	180.00

3	7.62512E-02	0	7.62512E-02	-2.23551E+01	7.62512E-02	0	7.62512E-02	0

SECOND ORDER: FREQUENCY(2, 2)= 4.650E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-3.55585E-03	-2.96051E-02	2.98179E-02	-3.05105E+01	-1.77793E-03	-1.48026E-02	1.49090E-02	-96.85
3	6.31191E-02	-6.91066E-02	9.35935E-02	-2.05751E+01	3.15595E-02	-3.45533E-02	4.67958E-02	-47.59

SECOND ORDER: FREQUENCY(2,-1)= 2.500E+05 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.32285E-05	-2.97124E-05	3.25242E-05	-8.97559E+01	1.32286E-05	-2.97124E-05	3.25242E-05	-65.00
3	8.12055E-02	-3.48884E-03	8.12804E-02	-2.18003E+01	8.12055E-02	-3.48884E-03	8.12804E-02	-2.46

SECOND ORDER: FREQUENCY(2,-2)= 0 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.05899E-14	0	1.05839E-14	-2.79502E+02	-1.05899E-14	0	1.05899E-14	180.00
3	8.66037E-02	0	8.66037E-02	-2.12493E+01	8.66037E-02	0	8.66037E-02	0

THIRD ORDER: FREQUENCY(1, 1, 1)= 6.900E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-3.44025E-02	3.05538E-02	4.60157E-02	-2.67419E+01	-8.60066E-03	7.63995E-03	1.15038E-02	138.39
3	-3.72443E-02	-1.30146E-02	3.94527E-02	-2.80785E+01	-9.31107E-03	-3.25366E-03	9.86318E-03	-160.74

THIRD ORDER: FREQUENCY(1, 1, 2)= 6.925E+07 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-3.49333E-02	3.44702E-02	4.91220E-02	-2.61745E+01	-2.62476E-02	2.58526E-02	3.68415E-02	135.43
3	-4.01310E-02	-1.18345E-02	4.18396E-02	-2.75682E+01	-3.00882E-02	-8.87598E-03	3.13797E-02	-163.57

THIRD ORDER: FREQUENCY(1, 1,-1)= 2.300E+07 HZ

TRANSFER FUNCTION OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.57602E-02	1.77800E-01	1.79656E-01	-1.49112E+01	-1.93202E-02	1.33350E-01	1.34742E-01	93.24
3	-4.24864E-01	-3.95346E-01	5.80352E-01	-4.72617E+00	-3.18648E-01	-2.96510E-01	4.35264E-01	-137.06

THIRD ORDER: FREQUENCY(1, 1,-2)= 2.275E+07 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-4.15843E-02	1.77226E-01	1.82039E-01	-1.47967E+01	-3.11883E-02	1.32919E-01	1.36529E-01	103.21
3	-3.91277E-01	-4.32673E-01	5.83355E-01	-4.68134E+00	-2.93458E-01	-3.24504E-01	4.37516E-01	-132.12

THIRD ORDER: FREQUENCY(1, 2, 2)= 6.950E+07 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-3.54152E-02	3.86659E-02	5.24344E-02	-2.56077E+01	-2.65614E-02	2.90002E-02	3.93253E-02	132.49
3	-4.31277E-02	-1.04345E-02	4.43720E-02	-2.70578E+01	-3.23458E-02	-7.82587E-03	3.32790E-02	-166.40

THIRD ORDER: FREQUENCY(1, 2,-1)= 2.325E+07 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.04931E-02	2.01461E-01	2.01734E-01	-1.39044E+01	-1.57471E-02	3.02192E-01	3.02602E-01	92.98
3	-5.20045E-01	-4.01564E-01	6.57033E-01	-3.64818E+00	-7.80067E-01	-6.02346E-01	9.85558E-01	-142.33

THIRD ORDER: FREQUENCY(1, 2,-2)= 2.300E+07 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.92060E-02	2.01948E-01	2.04043E-01	-1.38053E+01	-4.38090E-02	3.02922E-01	3.06074E-01	98.23
3	-4.82667E-01	-4.48907E-01	6.59154E-01	-3.62026E+00	-7.24000E-01	-6.73360E-01	9.88731E-01	-137.06

THIRD ORDER: FREQUENCY(2, 2, 2)= 6.975E+07 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG

2	-3.56312E-02	4.31591E-02	5.59669E-02	-2.50414E+01	-8.90781E-03	1.07898E-02	1.39917E-02	129.54
3	-4.62296E-02	-8.79537E-03	4.70589E-02	-2.65472E+01	-1.15574E-02	-2.19884E-03	1.17647E-02	-169.23

THIRD ORDER: FREQUENCY(2, 2,-1)= 2.350E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.03286E-02	2.26600E-01	2.26835E-01	-1.28858E+01	7.74649E-03	1.69950E-01	1.70127E-01	87.39
3	-6.31461E-01	-3.95342E-01	7.45009E-01	-2.55677E+00	-4.73596E-01	-2.96506E-01	5.58757E-01	-147.95

THIRD ORDER: FREQUENCY(2, 2,-2)= 2.325E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.18757E-02	2.28801E-01	2.29109E-01	-1.27992E+01	-8.90679E-03	1.71600E-01	1.71831E-01	92.97
3	-5.90709E-01	-4.55944E-01	7.46205E-01	-2.54283E+00	-4.43032E-01	-3.41958E-01	5.59654E-01	-142.34

INPUT FREQUENCIES:

FREQUENCY	VALUE(HZ)
1	4.300E+07
2	4.325E+07

FIRST ORDER: FREQUENCY(1)= 4.300E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-4.830935E-02	-3.03914E-02	5.70741E-02	-2.43712E+01	-4.830935E-02	-3.03914E-02	5.70741E-02	-147.83
3	5.17760E-01	-3.97568E-02	5.19284E-01	-5.69190E+00	5.17760E-01	-3.97568E-02	5.19284E-01	-4.39

FIRST ORDER: FREQUENCY(2)= 4.325E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-4.76550E-02	-2.96028E-02	5.61035E-02	-2.50193E+01	-4.76550E-02	-2.96028E-02	5.61035E-02	-148.15
3	5.14037E-01	-4.04198E-02	5.15674E-01	-5.75250E+00	5.14037E-01	-4.04198E-02	5.15674E-01	-4.50

SECOND ORDER: FREQUENCY(1, 1)= 8.600E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.48140E-03	-4.10337E-04	1.53748E-03	-5.62525E+01	7.40700E-04	-2.05444E-04	7.68748E-04	-15.50
3	1.48315E-03	1.72835E-04	1.49310E-03	-5.65177E+01	7.41574E-04	8.64175E-05	7.46555E-04	6.65

SECOND ORDER: FREQUENCY(1, 2)= 8.625E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.46101E-03	-4.10182E-04	1.51988E-03	-5.63634E+01	1.46101E-03	-4.10182E-04	1.51988E-03	-15.01
3	1.48328E-03	1.56337E-04	1.49143E-03	-5.65276E+01	1.48328E-03	1.56337E-04	1.49145E-03	6.02

SECOND ORDER: FREQUENCY(1,-1)= 0 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.66705E-15	0	2.66705E-15	-2.91479E+02	-2.66705E-15	0	2.66705E-15	180.00

3	3.54558E-03	0	3.54558E-03	-4.90063E+01	3.54558E-03	0	3.54558E-03	0

SECOND ORDER: FREQUENCY(2, 2)= 8.650E+07 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.44037E-03	-4.19392E-04	1.50066E-03	-5.64744E+01	7.20433E-04	-2.09696E-04	7.50330E-04	-16.23
3	1.48357E-03	1.40009E-04	1.49016E-03	-5.65354E+01	7.41783E-04	7.00045E-05	7.45079E-04	5.39

SECOND ORDER: FREQUENCY(2,-1)= 2.500E+05 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.57249E-07	-1.35254E-05	1.50378E-06	-1.16456E+02	6.57249E-07	-1.35254E-05	1.50378E-03	-84.03
3	3.50232E-03	1.11219E-05	3.50234E-03	-4.91128E+01	3.50232E-03	1.11219E-05	3.50234E-03	.18

SECOND ORDER: FREQUENCY(2,-2)= 0 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.62314E-15	0	2.62314E-15	-2.91624E+02	-2.62314E-15	0	2.62314E-15	180.00
3	3.45599E-03	0	3.45599E-03	-4.92285E+01	3.45599E-03	0	3.45599E-03	0

THIRD ORDER: FREQUENCY(1, 1, 1)= 1.290E+08 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	3.78248E-04	3.47002E-04	5.13379E-04	-6.57912E+01	9.45871E-05	8.67505E-05	1.28345E-04	42.53
3	-8.85553E-05	1.62492E-04	1.85253E-04	-7.46447E+01	-2.22415E-05	4.06230E-05	4.63133E-05	118.70

THIRD ORDER: FREQUENCY(1, 1, 2)= 1.293E+08 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	3.74955E-04	3.41040E-04	5.06854E-04	-6.59023E+01	2.81217E-04	2.55780E-04	3.80140E-04	42.29
3	-3.71743E-05	1.60465E-04	1.82616E-04	-7.47692E+01	-6.53807E-05	1.20349E-04	1.36963E-04	118.51

THIRD ORDER: FREQUENCY(1, 1,-1)= 4.300E+07 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.41012E-04	-3.67406E-04	3.93537E-04	-6.81003E+01	-1.05759E-04	-2.75554E-04	2.95153E-04	-111.00
3	7.15363E-04	-7.57775E-04	1.04210E-03	-5.96418E+01	5.36522E-04	-5.68331E-04	7.81573E-04	-46.66

THIRD ORDER: FREQUENCY(1, 1,-2)= 4.275E+07 HZ

TRANSFER FUNCTION — OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.40734E-04	-3.61729E-04	3.88141E-04	-6.82202E+01	-1.05550E-04	-2.71297E-04	2.91109E-04	-111.26
3	7.19361E-04	-7.57143E-04	1.04439E-03	-5.96228E+01	5.39520E-04	-5.67857E-04	7.83291E-04	-46.47

THIRD ORDER: FREQUENCY(1, 2, 2)= 1.295E+08 HZ

TRANSFER FUNCTION — OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	3.71582E-04	3.35168E-04	5.00411E-04	-6.60135E+01	2.78626E-04	2.51376E-04	3.75[illegible]E-04	42.05
3	-8.54171E-05	1.58456E-04	1.80021E-04	-7.48936E+01	-6.40628E-05	1.18849E-04	1.350[illegible]E-04	[illegible]

THIRD ORDER: FREQUENCY(1, 2,-1)= 4.325E+07 HZ

TRANSFER FUNCTION — OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.37836E-04	-3.63321E-04	3.89143E-04	-6.81977E+01	-2.06754E-04	-5.45881E-04	5.83724E-04	-110.74
3	6.94712E-04	-7.39409E-04	1.01457E-03	-5.98744E+01	1.04207E-03	-1.10911E-03	1.52188E-03	-46.78

THIRD ORDER: FREQUENCY(1, 2,-2)= 4.300E+07 HZ

TRANSFER FUNCTION — OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.37368E-04	-3.58153E-04	3.83593E-04	-6.83226E+01	-2.06053E-04	-5.37229E-04	5.75389E-04	-110.98
3	6.97472E-04	-7.38434E-04	1.01575E-03	-5.98642E+01	1.04621E-03	-1.10765E-03	1.52363E-03	-46.63

THIRD ORDER: FREQUENCY(2, 2, 2)= 1.298E+08 HZ

TRANSFER FUNCTION — OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG

2	3.68225E-04	3.29385E-04	4.94049E-04	-6.61246E+01	9.20563E-05	8.23462E-05	1.23512E-04	41.81
3	-8.36935E-05	1.56432E-04	1.77463E-04	-7.50177E+01	-2.09234E-05	3.91229E-05	4.43669E-05	118.14

THIRD ORDER: FREQUENCY(2, 2,-1)= 4.350E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.24335E-04	-3.60467E-04	3.34653E-04	-6.82940E+01	-1.01126E-04	-2.70350E-04	2.88644E-04	-110.51
3	6.74310E-04	-7.21613E-04	9.37975E-04	-6.01051E+01	5.06108E-04	-5.41210E-04	7.40981E-04	-46.92

THIRD ORDER: FREQUENCY(2, 2,-2)= 4.325E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.31107E-04	-3.54613E-04	3.79152E-04	-6.84237E+01	-1.00640E-04	-2.65930E-04	2.84334E-04	-110.70
3	6.70377E-04	-7.20301E-04	9.33063E-04	-6.01041E+01	5.07283E-04	-5.40226E-04	7.41063E-04	-46.80

INPUT FREQUENCIES:

FREQUENCY	VALUE(HZ)
1	6.300E+07
2	6.325E+07

FIRST ORDER: FREQUENCY(1)= 6.300E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.05059E-02	-5.37373E-03	2.11983E-02	-3.34740E+01	-2.05059E-02	-5.37373E-03	2.11983E-02	-165.32
3	3.82162E-01	-6.53621E-02	3.87711E-01	-8.22984E+00	3.82162E-01	-6.53621E-02	3.87711E-01	-9.71

FIRST ORDER: FREQUENCY(2)= 6.325E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-2.03589E-02	-5.30180E-03	2.10379E-02	-3.35399E+01	-2.03589E-02	-5.30180E-03	2.10379E-02	-165.40
3	3.81399E-01	-6.55743E-02	3.86992E-01	-8.24595E+00	3.81399E-01	-6.55743E-02	3.86992E-01	-9.72

SECOND ORDER: FREQUENCY(1, 1)= 1.250E+03 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.85703E-04	-2.65579E-04	3.90075E-04	-6.81770E+01	1.42851E-04	-1.32789E-04	1.95037E-04	-42.91
3	1.41430E-03	-1.01825E-03	1.74313E-03	-5.51734E+01	7.07402E-04	-5.09125E-04	8.71559E-04	-35.74

SECOND ORDER: FREQUENCY(1, 2)= 1.263E+08 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.83331E-04	-2.64674E-04	3.87722E-04	-6.82296E+01	2.83331E-04	-2.64674E-04	3.87722E-04	-43.05
3	1.41321E-03	-1.02200E-03	1.74403E-03	-5.51689E+01	1.41321E-03	-1.02200E-03	1.74403E-03	-35.87

SECOND ORDER: FREQUENCY(1,-1)= 0 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-8.31112E-16	0	8.31112E-16	-3.01607E+02	-8.31112E-16	0	8.31112E-16	180.00

3	9.13124E-04	0	9.13124E-04	-6.07894E+01	9.13124E-04	0	9.13124E-04	0

SECOND ORDER: FREQUENCY(2, 2)= 1.265E+08 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	2.80975E-04	-2.63768E-04	3.85383E-04	-6.82821E+01	1.40487E-04	-1.31884E-04	1.92692E-04	-43.19
3	1.41161E-03	-1.02574E-03	1.74493E-03	-5.51644E+01	7.05805E-04	-5.12871E-04	8.72466E-04	-35.00

SECOND ORDER: FREQUENCY(2,-1)= 2.500E+05 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.68576E-08	-3.33644E-07	3.34069E-07	-1.29523E+02	1.68576E-08	-3.33644E-07	3.34069E-07	-87.11
3	9.08266E-04	5.12523E-06	9.08280E-04	-6.08356E+01	9.08266E-04	5.12523E-06	9.08280E-04	.32

SECOND ORDER: FREQUENCY(2,-2)= 0 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-8.20584E-16	0	8.20584E-16	-3.01718E+02	-8.20584E-16	0	8.20584E-16	180.00
3	9.02404E-04	0	9.02404E-04	-6.08920E+01	9.02404E-04	0	9.02404E-04	0

THIRD ORDER: FREQUENCY(1, 1, 1)= 1.890E+08 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.49896E-05	-2.63355E-05	6.50430E-05	-8.37360E+01	1.62474E-05	-6.58413E-07	1.62607E-05	-2.32
3	2.98444E-06	2.66936E-05	2.68599E-05	-9.14179E+01	7.46111E-07	6.67339E-06	6.71497E-06	83.62

THIRD ORDER: FREQUENCY(1, 1, 2)= 1.893E+08 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	5.45853E-05	-2.79758E-06	6.46458E-05	-8.37892E+01	4.84390E-05	-2.09818E-06	4.84844E-05	-2.48
3	3.03281E-06	2.65831E-05	2.67556E-05	-9.14517E+01	2.27461E-06	1.99373E-05	2.00667E-05	83.49

THIRD ORDER: FREQUENCY(1, 1,-1)= 6.300E+07 HZ

TRANSFER FUNCTION / OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.94210E-05	-5.54933E-05	5.87973E-05	-8.46128E+01	-1.45733E-05	-4.16203E-05	4.40980E-05	[illegible]
3	2.05506E-05	-4.16511E-05	4.64451E-05	-8.66612E+01	1.54130E-05	-3.12083E-05	[illegible]	-63.74

THIRD ORDER: FREQUENCY(1, 1,-2)= 6.275E+07 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.93502E-05	-5.50321E-05	5.83018E-05	-8.46864E+01	-1.44276E-05	-4.12741E-05	4.37283E-05	[illegible]
3	2.07022E-05	-4.13009E-05	4.96467E-05	-8.66236E+01	1.55270E-05	-3.13507E-05	3.49850E-05	[illegible]

THIRD ORDER: FREQUENCY(1, 2, 2)= 1.835E+03 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	6.41229E-05	-2.55333E-05	6.42511E-05	-8.38424E+01	4.81372E-05	-2.21950E-06	[illegible]	[illegible]
3	3.00078E-05	2.54734E-05	2.56521E-05	-9.14854E+01	2.31059E-06	1.98551E-05	[illegible]	[illegible]

THIRD ORDER: FREQUENCY(1, 2,-1)= 5.325E+07 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.94053E-05	-5.53419E-05	5.86453E-05	-8.46353E+01	-2.91093E-05	-8.30128E-05	[illegible]	[illegible]
3	2.01572E-05	-4.10458E-05	4.57283E-05	-8.67963E+01	3.02358E-05	-6.15988E-05	6.85924E-05	[illegible]

THIRD ORDER: FREQUENCY(1, 2,-2)= 6.300E+07 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.91995E-05	-5.43453E-05	5.81093E-05	-8.47151E+01	-2.87993E-05	-8.22689E-05	8.71640E-05	-109.29
3	2.01033E-05	-4.10952E-05	4.57513E-05	-8.67919E+01	3.01634E-05	-6.16427E-05	[illegible]	-63.93

THIRD ORDER: FREQUENCY(2, 2, 2)= 1.897E+08 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG

2	6.37825E-05	-3.11893E-06	6.38587E-05	-8.38956E+01	1.59456E-05	-7.79733E-07	1.59647E-05	-2.80
3	3.12836E-06	2.63645E-05	2.65494E-05	-9.15189E+01	7.82091E-07	6.59112E-06	6.63736E-06	83.23

THIRD ORDER: FREQUENCY(2, 2,-1)= 6.350E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.93954E-05	-5.51990E-05	5.85077E-05	-8.46557E+01	-1.45473E-05	-4.13992E-05	4.38808E-05	-109.36
3	1.97707E-05	-4.04439E-05	4.50222E-05	-8.69315E+01	1.48280E-05	-3.03367E-05	3.37666E-05	-63.95

THIRD ORDER: FREQUENCY(2, 2,-2)= 6.325E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.91608E-05	-5.43707E-05	5.79312E-05	-3.47417E+01	-1.43707E-05	-4.10020E-05	4.34434E-05	-109.31
3	1.95240E-05	-4.00331E-05	4.43688E-05	-8.69311E+01	1.46433E-05	-3.02986E-05	3.36516E-05	-64.21

INPUT FREQUENCIES:

FREQUENCY	VALUE(HZ)
1	8.300E+07
2	8.325E+07

FIRST ORDER: FREQUENCY(1)= 8.300E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.16963E-02	-4.50892E-05	1.16964E-02	-3.86390E+01	-1.16963E-02	-4.50892E-05	1.16964E-02	-179.78
3	3.41029E-01	-8.15271E-02	3.50638E-01	-9.10281E+00	3.41029E-01	-8.15271E-02	3.50638E-01	-13.44

FIRST ORDER: FREQUENCY(2)= 8.325E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.16173E-02	-1.54350E-05	1.16174E-02	-3.86979E+01	-1.16173E-02	-1.54350E-05	1.16174E-02	-179.92
3	3.40672E-01	-8.17216E-02	3.50336E-01	-9.11030E+00	3.40672E-01	-8.17216E-02	3.50336E-01	-13.49

SECOND ORDER: FREQUENCY(1, 1)= 1.660E+08 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	7.66070E-05	-1.59451E-04	1.76899E-04	-7.50455E+01	3.83035E-05	-7.97255E-05	8.84495E-05	-64.34
3	1.06890E-03	-1.41614E-03	1.77426E-03	-5.50197E+01	5.34445E-04	-7.08068E-04	8.87128E-04	-52.95

SECOND ORDER: FREQUENCY(1, 2)= 1.662E+08 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	7.59830E-05	-1.58350E-04	1.76157E-04	-7.50315E+01	7.59830E-05	-1.58930E-04	1.76157E-04	-64.47
3	1.06640E-03	-1.41763E-03	1.77395E-03	-5.50212E+01	1.06640E-03	-1.41763E-03	1.77395E-03	-53.05

SECOND ORDER: FREQUENCY(1,-1)= 0 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-3.34320E-16	0	3.34320E-16	-3.09517E+02	-3.34320E-16	0	3.34320E-16	180.00

3	4.25217E-04	0	4.25217E-04	-6.74278E+01	4.25217E-04	0	4.25217E-04	0

SECOND ORDER: FREQUENCY(2, 2)= 1.565E+08 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	7.52723E-05	-1.58470E-04	1.75433E-04	-7.51175E+01	3.76364E-05	-7.92349E-05	8.77199E-05	-64.59
3	1.06390E-03	-1.41912E-03	1.77364E-03	-5.50227E+01	5.31950E-04	-7.09560E-04	8.86810E-04	-53.14

SECOND ORDER: FREQUENCY(2,-1)= 2.500E+05 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.08379E-07	-1.36037E-07	1.73931E-07	-1.35192E+02	-1.08379E-07	-1.36037E-07	1.72931E-07	-128.54
3	4.23733E-04	3.30224E-06	4.23746E-04	-6.74579E+01	4.23733E-04	3.30224E-06	4.23746E-04	.45

SECOND ORDER: FREQUENCY(2,-2)= 0 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-3.30951E-16	0	3.30951E-16	-3.09605E+02	-3.30951E-16	0	3.30951E-16	180.00
3	4.21860E-04	0	4.21860E-04	-6.74966E+01	4.21860E-04	0	4.21860E-04	0

THIRD ORDER: FREQUENCY(1, 1, 1)= 2.490E+08 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.50721E-05	-1.21057E-05	1.93318E-05	-9.42746E+01	3.76803E-06	-3.02643E-06	4.83295E-06	-38.77
3	9.09123E-06	1.24641E-05	1.54274E-05	-9.62341E+01	2.27282E-06	3.11602E-06	3.85683E-06	53.89

THIRD ORDER: FREQUENCY(1, 1, 2)= 2.493E+08 HZ

PORT NO	TRANSFER FUNCTION REAL	IMAGINARY	MAGNITUDE	20LOG MAG	OUTPUT VOLTAGE REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.49771E-05	-1.20919E-05	1.92491E-05	-9.43118E+01	1.12228E-05	-9.06890E-06	1.44290E-05	-38.94
3	9.10429E-06	1.24283E-05	1.54062E-05	-9.62461E+01	6.82222E-06	9.32119E-06	1.15511E-05	53.80

THIRD ORDER: FREQUENCY(1, 1,-1)= 8.300E+07 HZ

TRANSFER FUNCTION OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.07529E-05	-1.73825E-05	2.04395E-05	-9.37906E+01	-8.06457E-06	-1.30369E-05	1.53297E-05	-121.74
3	-1.10115E-05	2.24383E-06	1.12378E-05	-9.89864E+01	-8.25862E-06	1.68287E-06	8.42034E-06	168.43

THIRD ORDER: FREQUENCY(1, 1,-2)= 8.275E+07 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.06341E-05	-1.73329E-05	2.03510E-05	-9.38240E+01	-8.01311E-06	-1.29994E-05	1.52707E-05	-121.65
3	-1.10022E-05	2.12464E-06	1.12054E-05	-9.90114E+01	-8.25163E-06	1.59348E-06	8.40408E-06	169.07

THIRD ORDER: FREQUENCY(1, 2, 2)= 2.495E+03 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	1.48325E-05	-1.20778E-05	1.91667E-05	-9.43490E+01	1.11619E-05	-9.05337E-06	1.43751E-05	-39.03
3	9.11724E-06	1.23925E-05	1.53850E-05	-9.62580E+01	6.83793E-06	9.29440E-06	1.15380E-05	53.66

THIRD ORDER: FREQUENCY(1, 2,-1)= 8.325E+07 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.07545E-05	-1.73030E-05	2.03723E-05	-9.38189E+01	-1.61319E-05	-2.59545E-05	3.05534E-05	-121.86
3	-1.09831E-05	2.35197E-06	1.12321E-05	-9.89907E+01	-1.64747E-05	3.52795E-06	1.68482E-05	167.91

THIRD ORDER: FREQUENCY(1, 2,-2)= 8.300E+07 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.06783E-05	-1.72435E-05	2.02824E-05	-9.38576E+01	-1.60182E-05	-2.58552E-05	3.04235E-05	-121.77
3	-1.10580E-05	2.28945E-06	1.12925E-05	-9.89442E+01	-1.65870E-05	3.43418E-06	1.69388E-05	168.30

THIRD ORDER: FREQUENCY(2, 2, 2)= 2.498E+08 HZ

TRANSFER FUNCTION | OUTPUT VOLTAGE

PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG

2	1.47884E-05	-1.20635E-05	1.90847E-05	-9.43863E+01	3.69711E-06	-3.01590E-05	4.77119E-05	-39.21
3	9.13013E-06	1.23569E-05	1.53640E-05	-9.62699E+01	2.28253E-06	3.08922E-05	3.84109E-05	53.54

THIRD ORDER: FREQUENCY(2, 2,-1)= 8.350E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.07551E-05	-1.72237E-05	2.03054E-05	-9.33474E+01	-8.06706E-06	-1.29178E-05	1.52[illegible]E-05	-121.98
3	-1.09345E-05	2.45907E-06	1.12271E-05	-9.89947E+01	-8.21585E-06	1.84430E-06	8.420[illegible]E-06	167.35

THIRD ORDER: FREQUENCY(2, 2,-2)= 8.325E+07 HZ

	TRANSFER FUNCTION				OUTPUT VOLTAGE			
PORT NO	REAL	IMAGINARY	MAGNITUDE	20LOG MAG	REAL	IMAGINARY	MAGNITUDE	PHASE DEG
2	-1.06733E-05	-1.71546E-05	2.02033E-05	-9.38313E+01	-8.00493E-06	-1.28660E-05	1.5[illegible]E-05	-121.89
3	-1.11130E-05	2.45317E-06	1.13805E-05	-9.83768E+01	-8.33472E-06	1.83380E-06	8.5[illegible]E-06	167.55

TIME FOR FORMING ZCC(SEC) 3.9170
TIME FOR OBTAINING OUTPUT SPECTRUM(SEC) 13.0210
TOTAL EXECUTION TIME(SEC) 21.9580

*** P R A N C ***
SEPTEMBER 1979 VERSION

CHAPTER 5

PROGRAMMER'S GUIDE FOR PRANC

5-1. Introduction

The ideas presented in Chapters 2 and 3 have been used to adapt the Volterra series method for computer-aided distortion analysis of circuits with polynomial type nonlinear elements. PRANC (Program for Analyzing Nonlinear Circuits), a digital computer program written in FORTRAN IV, is the outcome of this effort. This chapter presents in detail the program structure and the description of the subprograms contained in PRANC. This chapter should be most useful for programmers wishing to modify the program.

Section 5-2 presents the program structure of PRANC. By pointing out the sequence of phases that are involved in a typical analysis run, the interaction between the various subprograms is depicted. The discussion in this section provides an insight into how the "equivalencing" of arrays should be carried out for conserving storage.

Section 4-3 presents the details of each subprogram contained in PRANC. These details include: 1) brief description; 2) glossary of FORTRAN variables; and 3) listing of each subprogram. The contents of this section should aid the programmer in making future modifications to the program.

PRANC has been developed on the CDC 6500/6600 computer at the Purdue University computing facility, and as such certain machine- and library-dependent instructions exist. These system dependent cards in the program are listed in section 5-4. Cards capable of calling equivalent functions can be substituted in their place for adapting the program on a different system.

5-2. Program Structure of PRANC

Before detailing the program structure of PRANC, it is instructive to delineate the sequence of steps that are involved in a typical analysis run when using our computational algorithms. The program structure and its modularity are better understood once a knowledge of the sequence of steps has been acquired. Referring to a collection of steps as a phase, the following is a partitioning of phases that are involved in a typical analysis run:

<u>Phase A</u>: The following functions are performed during this phase:

(a) Read input data

(b) Control the interaction between the other phases.

In a sense, this phase can be regarded to extend during the entire analysis run.

<u>Phase B</u>: The user desired options are scanned during this phase and the flag variable associated with each option is appropriately set.

<u>Phase C</u>: This phase is responsible for the following functions:

(a) Setting up of the arrays for the network description in a prescribed manner.

(b) Assigning addresses based on the user-specified options and the nonlinear element topology.

<u>Phase D</u>: The Hybrid analysis, which yields the constraint matrix [20], is performed during this phase.

<u>Phase E</u>: The state space representation for the linearized circuit is obtained during this phase.

Phase F: The eigenvalue-eigenvector information is determined from the state space description during this phase.

Phase G: The printing and the formation of the entries of the open-circuit impedance matrix is carried out during this phase.

Phase H: The first-, second-, and third-order transfer functions are computed during this phase.

Phase I: The following functions are performed during this phase:

(a) Compute the output voltages at each frequency point from the transfer function values.

(b) Print both the transfer function and the output voltage values at each discrete frequency point for the requested outputs.

Phase J: During this phase, the complete output spectrum at the user-requested port is printed and plotted.

Phase K: When a frequency sweep capability is requested, this phase is used to perform the said operation.

Phase L: When devices, such as transistors, diodes, etc., are to be represented by equivalent nonlinear models, this phase is used to calculate the parameters of the nonlinearities.

Several subroutines are required to perform the functions belonging to each of the aforementioned phases. PRANC, in its present version, consists of thirty-six sub-programs, whose interaction is depicted in Fig. 5-1. In order to provide a link between a phase and its associated sub-programs, the naming of the subroutines has been done in a deterministic manner: the first letter of the subroutine name signifies the phase to which it belongs.

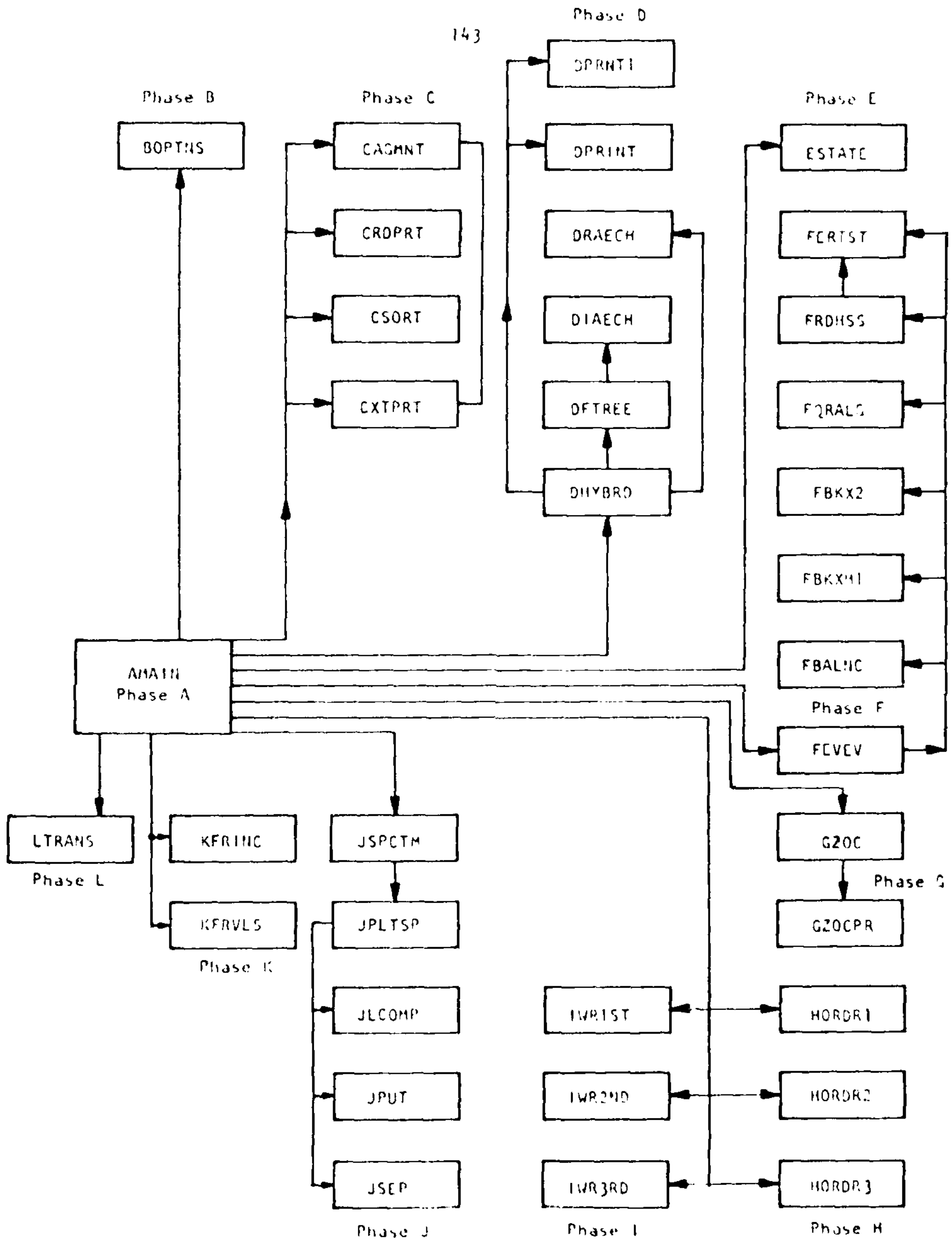


Fig. 5-1. Program Structure of PRANC

Thus, for example, the subroutines HORDR1, HORDR2, HORDR3 perform the functions outlined under phase H.

In the following paragraphs the function of each subprogram appearing in Fig. 5-1 is outlined:

Program AMAIN is the executive calling program of PRANC.

Subroutine BOPTNS deciphers the desired user options.

Subroutine CAGMNT forms the augmented linear network by lumping the linear parts of the nonlinear elements with the existing linear network.

Subroutine CRDPRT identifies and combines the parallel energy storage elements and current sources appearing in the augmented linear network, thus effectively reducing the number of ports extracted for hybrid analysis.

Subroutine CSORT sorts the elements of the augmented linear network and arranges them in an order suitable for choosing a proper tree [20].

Subroutine CXTPRT adds a branch to the linear network.

Subroutine DFTREE finds the proper tree from the incidence matrix [20].

Subroutine DHYBRD is the executive calling program for performing hybrid analysis of the augmented linear circuit to obtain the constraint matrix.

Subroutine DIAECH is used to manipulate the incidence matrix into echelon form.

Subroutine DPRINT prints the entire constraint matrix whenever the debug option is requested by the user.

Subroutine DPRNT1 prints only part of the constraint matrix describing the port equations whenever the debug option is requested by the user.

Subroutine DRAECH operates on the rows of the hybrid matrix to reduce it into echelon form.

Subroutine ESTATE formulates the state space description of the augmented linear network and, if desired by the user, prints this description.

Subroutine FBALNC balances the matrix whose eigenvalues are to be determined.

Subroutine FEVEV is the executive calling program used to determine the eigenvalues and their associated eigenvectors.

Subroutine FBKXM1 is used to back transform the eigenvectors of an Hessenberg matrix.

Subroutine FBKXM2 is used to back-transform the eigenvectors of a balanced matrix.

Subroutine FERTST is used to print the error diagnosis arising in eigenvalues-eigenvectors problems.

Subroutine FQRALG determines the eigenvalues and the eigenvectors of the Hessenberg matrix.

Subroutine FRDHSS reduces a matrix to the Hessenberg form.

Subroutine GZOC forms the matrices used to store the entries of the open circuit impedance matrix.

Subroutine GZOCPR prints the entries of the open circuit impedance matrix whenever desired by the user.

Subroutine HORDR1 computes the first-order transfer function at each positive and negative input frequency value.

Subroutine HORDR2 computes the second-order transfer function at each frequency combination appearing in the second-order output spectrum.

Subroutine HORDR3 computes the third-order transfer function at each non-negative frequency combination appearing in the third-order output spectrum.

Subroutine IWR1ST determines the first-order output spectrum and prints it along with the first-order transfer function at the user-specified output ports.

Subroutine IWR2ND determines the second-order output spectrum for non-negative frequencies and prints it along with the second-order transfer function values at the user-specified output ports.

Subroutine IWR3RD determines the third-order output spectrum for non-negative frequencies and prints it along with the third-order transfer function values at the user-specified output ports.

Subroutine JSPCTM performs histogram analysis of all output frequency components and combines the common-ones. It also prints and plots the complete output spectrum at the user-requested port, whenever desired.

Subroutine JPLTSP perform the actual plotting of the output spectrum.

Function JLCOMP locates the data points for plotting.

Function JPUT also locates the data points for plotting.

Subroutine JSEP separates the alphabets in the y-axis label for vertical printing.

Subroutine KFRNC computes the frequency increments for each input frequency whenever the frequency sweep capability is requested.

Subroutine KFRVLS computes the new frequency values during the frequency sweep.

Subroutine LTRANS computes the coefficients of the polynomials representing the nonlinear elements in a bipolar transistor.

5-3. Glossary and Subprogram Listings for PRANC

In this section we shall present the specific task of each sub-program, along with its listing. The glossary of important FORTRAN variable names is included in the sub-program listing.

5-3.1 Program AMAIN

Program AMAIN is the executive calling program of PRANC. Its primary function is to read and write input data, to form appropriate arrays for the augmented linear network description, and to assign appropriate addresses for subsequent use in forming the nonlinear current sources. The addressing array NCONT used in PRANC deserves some explanation.

Based on the network element types, and their associated KEY values, the elements of the augmented linear network are arranged in the following order:

1. Capacitors
2. VCVSs
3. CCVSs
4. Resistors
5. Inductors
6. VCCSs
7. CCCSs
8. Independent current sources

Such an arrangement is warranted for the selection of a proper tree and the formulation of hybrid and state equations. The number of independent current sources is equal to the number of extracted ports, p, for the augmented linear network.

Initially, as each input information card is read, a zero-valued current source is applied across each prescribed input source branch, output branch, nonlinear element branch, and nonlinear element characteristic controlling branch (in the dependent nonlinear element case). Each zero-valued current source signifies an extracted port. Associated with each extracted port is an index number, NCONT, starting from 1 to n ($n \geq p$). Clearly some of the initially extracted ports may be in parallel. The p-port augmented linear network is obtained after the parallel zero-valued current source branches in the n-port network are combined.

The extracted ports in the p-ports network has the following arrangement:

Input port	NCONT(1)
Output port 1	NCONT(2)
Output port 2	NCONT(3)
•	
•	
•	
Output port k	NCONT(k+1)
Nonlinear element #1 port	NCONT(k+2)
Nonlinear element #1 controlling port	
Nonlinear element #1 controlling port	
Nonlinear element #2 port	
Nonlinear element #2 controlling port	
Nonlinear element #2 controlling port	
•	
•	
•	
Nonlinear element #ℓ port	
Nonlinear element #ℓ controlling port	
Nonlinear element #ℓ controlling port	NCONT(3ℓ+k+1)

The array NCONT contains the port number for each of the above extracted ports. Thus, NCONT(1) contains the port number for the input source port; NCONT(2) for the first output port; NCONT(3) for the second output port; NCONT(k+1) for the k-th output port; and so on. It should be noted that the independent nonlinear elements are treated as special cases of dependent nonlinear elements. Thus, if NCONT(5) = 3 signifies port number 3 for a nonlinear capacitor, then the locations NCONT(6) and NCONT(7) will also contain 3. In the case of a dependent nonlinear element, the number for the controlling ports will usually be different. It should be clear from the above discussion that, for a single input, k-output network with ℓ nonlinearities, the length of the array used is $(3\ell+k+1)$. The array NCONT plays an important role when the various order steady-state responses are computed.

5-3.2. PRANC Listing:

```
C**********************************************************************AMN
C**************************** P R A N C *******************************AMN
C************** PROGRAM FOR ANALYZING NONLINEAR CIRCUITS ***************AMN
C**********************************************************************AMN
C********************** SEPTEMBER 1979 VERSION ************************AMN
C**********************************************************************AMN
C**********************************************************************AMN
      PROGRAM AMAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)              AMN
C**********************************************************************AMN
C     *                                                                *AMN
C*****  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTIONS:             *AMN
C     *       1. READ AND WRITE INPUT CIRCUIT DESCRIPTION.             *AMN
C     *       2. ACT AS THE EXECUTIVE CALLING PROGRAM FOR PRANC.       *AMN
C     *                                                                *AMN
C*****  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINES:               *AMN
C     *       1. DOPTNS                                                *AMN
C     *       2. CASMNT,CKTPRT,CRDPRT,CSORT                            *AMN
C     *       3. BHYBRD                                                *AMN
C     *       4. ESTATE                                                *AMN
C     *       5. FEVEV                                                 *AMN
C     *       6. GEOC                                                  *AMN
C     *       7. HORDR1, HORDR2, HORDR3                                *AMN
C     *       8. INR1ST, INR2ND, INR3RD                                *AMN
C     *       9. JEPCTM                                                *AMN
C     *      10. KFRINC, KFRULS                                        *AMN
C     *      11. LTRANS                                                *AMN
C     *      12.**** SECOND ****(LIBRARY DEPENDENT ROUTINE)            *AMN
C     *                                                                *AMN
C*****  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                  *AMN
C     *       BR(K)      : BRANCH NUMBER K IN THE AUGMENTED LINEAR     *AMN
C     *                    NETWORK                                     *AMN
C     *       NFROM(K)   : ~FROM~(+) NODE NUMBER FOR BRANCH NUMBER K   *AMN
C     *       NTO(K)     : ~TO~(-) NODE NUMBER FOR BRANCH NUMBER K     *AMN
C     *       TYPE(K)    : BRANCH NUMBER K ELEMENT TYPE                *AMN
C     *       ICONT(K)   : CONTROLLING BRANCH NUMBER FOR BRANCH K      *AMN
C     *       KEY(K)     : KEY VALUE FOR BRANCH K                      *AMN
C     *       VALUE(K)   : ELEMENT VALUE FOR BRANCH K                  *AMN
C     *       NTYPE(K)   : TYPE OF NONLINEAR ELEMENT K                 *AMN
C     *       COFF(K,J)  : POLYNOMIAL COEFFICIENT VALUE FOR NONLINEAR  *AMN
C     *                    ELEMENT K                                   *AMN
C     *       FREQ(I)    : I-TH INPUT FREQUENCY VALUE                  *AMN
C     *       AMP(I)     : I-TH INPUT FREQUENCY AMPLITUDE              *AMN
C     *       PHASE(I)   : I-TH INPUT FREQUENCY PHASE                  *AMN
C     *       LUNIT      : INPUT FREQUENCY UNIT (HZ OR RAD/SEC)        *AMN
C     *       A          : INCIDENCE MATRIX FOR THE AUGMENTED LINEAR   *AMN
C     *                    NETWORK                                     *AMN
C     *       AMC        : CONSTRAINT (HYBRID) MATRIX                  *AMN
C     *       HEADER     : HEADING VECTOR FOR THE CONSTRAINT MATRIX    *AMN
C     *       EXTR1      : WORK MATRIX USED IN HYBRID ANALYSIS         *AMN
C     *       EXTR2      : WORK MATRIX USED IN HYBRID ANALYSIS         *AMN
C     *       AMAT       : MATRIX A OF THE STATE SPACE REPRESENTATION  *AMN
C     *       BMAT       : MATRIX B OF THE STATE SPACE REPRESENTATION  *AMN
C     *       CMAT       : MATRIX C OF THE STATE SPACE REPRESENTATION  *AMN
C     *       DMAT       : MATRIX D OF THE STATE SPACE REPRESENTATION  *AMN
C     *       EVALS(I)   : I-TH COMPLEX EIGENVALUE (NATURAL FREQUENCY) *AMN
C     *       EVECTS     : MODAL MATRIX FOR SIMILARITY TRANSFORMATION  *AMN
C     *       BHAT       : MATRIX OBTAINED FROM THE PRODUCT OF MODAL   *AMN
C     *                    MATRIX INVERSE AND BMAT                     *AMN
C     *       CHAT       : MATRIX OBTAINED FROM THE PRODUCT OF CMAT AND *AMN
C     *                    MODAL MATRIX                                *AMN
```

```
C       *         WK1,WK2    : WORK ARRAYS                                      *
C       *         NSTPS(I)   : I-TH INPUT FREQUENCY NUMBER OF INCREMENTS        *
C       *         FRINC(I)   : I-TH INPUT FREQUENCY INCREMENT VALUE             *
C       *         HFR(I)     : I-TH INPUT FREQUENCY HIGHEST VALUE               *
C       *         Y1(P,I)    : PORT P FIRST-ORDER OUTPUT AT FREQ(I)             *
C       *         W2(I)      : SECOND-ORDER I-TH FREQUENCY COMPONENT VALUE      *
C       *         Y2(P,I)    : PORT P SECOND-ORDER OUTPUT AT W2(I)              *
C       *         FC2(I)     : COMBINATION CODE FOR W2(I)                       *
C       *         W3(I)      : THIRD-ORDER I-TH FREQUENCY COMPONENT VALUE       *
C       *         Y3(P,I)    : PORT P THIRD-ORDER OUTPUT AT W3(I)               *
C       *         FC3(I)     : FREQUENCY COMBINATION CODE FOR W3(I)             *
C       *         FR(I)      : I-TH FREQUENCY VALUE IN THE COMPLETE SPECTRUM*
C       *         Y(I)       : OUTPUT VOLTAGE AT FR(I)                          *
C       *         IPT        : ARRAY USED FOR ~HISTOGRAM~ ANALYSIS              *
C       *         YLG        : LOG OF THE OUTPUT (Y), USED FOR PLOTTING         *
C       *         ST1,ST2    : DUMMY STORAGE ARRAYS USED FOR EQUIVALENCING      *
C       *         NCONT      : ARRAY FOR ADDRESSING NONLINEAR CURRENT           *
C       *                      SOURCES AND REQUESTED OUTPUT PORTS (SEE          *
C       *                      TECHNICAL REPORT FOR DETAILS)                    *
C       *         JCONT(K)   : SECOND CONTROLLING BRANCH NUMBER FOR NONLINEA*
C       *                      ELEMENT K/ SUBSEQUENTLY IDENTIFIES THE           *
C       *                      NONLINEAR ELEMENT TYPE                           *
C       *         TITLE      : ARRAY USED FOR READING TITLE AND OPTION CARD *
C       *         NCAP       : NUMBER OF LINEAR CAPACITORS                      *
C       *         NDVS       : NUMBER OF LINEAR DEPENDENT VOLTAGE SOURCES       *
C       *         NRES       : NUMBER OF LINEAR RESISTORS                       *
C       *         NIND       : NUMBER OF LINEAR INDUCTORS                       *
C       *         NDCS       : NUMBER OF LINEAR DEPENDENT CURRENT SOURCES       *
C       *         NCS        : NUMBER OF LINEAR CURRENT SOURCES(=# OF PORTS)*
C       *                                                                       *
C***********************************************************************
C
C
      INTEGER A,BR,TYPE,ANSCOL,FROM,TO,HEADER,OUTPT
      INTEGER R,G,C,E,CV,VV,CC,VC,TITLE
      INTEGER DB,SE,FS,PR,PC,AP
      COMPLEX TH
C
C***** ARRAYS REQUIRED FOR STORING AUGMENTED LINEAR NETWORK
C
      DIMENSION BR(75), NFROM(75), NTO(75), TYPE(75), ICONT(75), KEY(75)
C
C***** ARRAY FOR ELEMENT VALUES
C
      DIMENSION VALUE(75)
C
C***** ARRAYS FOR NONLINEAR ELEMENT TYPE AND POLYNOMIAL COEFFICIENTS
C
      COMMON /001/ NTYPE(10),COFF(10,9)
C
C***** INPUT AMPLITUDE AND FREQUENCY ARRAYS
C
      COMMON /003/ FREQ(10),AMP(10),TH(10),LUNIT
      DIMENSION PHASE(5)
C
C***** ARRAYS FOR HYBRID ANALYSIS
C
      DIMENSION A(30,75), ANS(75,150), HEADER(300), EXTR1(75,30), EXTR2(
     175,30)
C
```

```
C***** ARRAYS FOR THE FORMATION OF STATE EQUATIONS
C
      DIMENSION AMAT(20,20), BMAT(20,20), CMAT(25,20), DMAT(25,25)
C
C***** EIGENVALUE AND EIGENVECTOR ARRAYS
C
      COMPLEX EVALS(20),EVECTS(20,20)
C
C***** ARRAYS FOR STORING PFE INFO AND WORK ARRAYS
C
      COMPLEX DHAT(20,25),CHAT(25,20),WK1(20,20),WK2(20,20)
C
C***** ARRAYS FOR FREQUENCY SWEEP
C
      COMMON /004/ NSTPS(5),FRINC(5),HFR(5)
C
C***** ARRAYS FOR FIRST-ORDER TRANSFER FUNCTIONS
C
      COMPLEX Y1(25,10)
C
C***** ARRAYS FOR SECOND-ORDER TRANSFER FUNCTIONS
C
      COMPLEX Y2(25,55),WKZ(25,25)
      DIMENSION W2(55)
      INTEGER FC2(55)
C
C***** ARRAYS FOR THIRD-ORDER TRANSFER FUNCTIONS
C
      COMPLEX Y3(25,130)
      DIMENSION W3(130)
      INTEGER FC3(130)
C
C***** ARRAYS FOR COMPLETE OUTPUT SPECTRUM
C
      COMPLEX Y(160)
      DIMENSION FR(160), IPT(160), YLG(160)
C
C***** MISCELLANEOUS WORK ARRAYS
C
      DIMENSION ST1(75,100), ST2(50,225), NLBN(32), TITLE(80), NPORT(25)
      COMMON /016/ NCONT(32),JCONT(10)
C
      COMMON /ETYPE/ R,G,L,C,E,IS,CV,VV,CC,VC
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NDCS,NCS
C
C***** EQUIVALENCE FOR PHASE 1 (OBTAIN HYBRID MATRIX)
C
      EQUIVALENCE (ST1(1),TITLE(1))
      EQUIVALENCE (ST1(1),DR(1))
      EQUIVALENCE (ST1(76),NFROM(1))
      EQUIVALENCE (ST1(151),NTO(1))
      EQUIVALENCE (ST1(226),TYPE(1))
      EQUIVALENCE (ST1(301),ICONT(1))
      EQUIVALENCE (ST1(376),VALUE(1))
      EQUIVALENCE (ST1(451),KEY(1))
      EQUIVALENCE (ST1(526),EXTR1(1))
      EQUIVALENCE (ST1(2776),EXTR2(1))
      EQUIVALENCE (ST1(5026),A(1))
      EQUIVALENCE (ST1(7276),HEADER(1))
      EQUIVALENCE (ST2(1),ANS(1))
```

```
C
C***** EQUIVALENCE FOR PHASE 2 (OBTAIN STATE EQUATIONS)
C
      EQUIVALENCE (ST1(S23),AMAT(1))
      EQUIVALENCE (ST1(1013),DMAT(1))
      EQUIVALENCE (ST1(1003),CMAT(1))
      EQUIVALENCE (ST1(1),EMAT(1))
C
C***** EQUIVALENCE FOR PHASE 3 (OBTAIN Z(S) IN PFE FORM)
C
      EQUIVALENCE (ST2(2001),UK1(1))
      EQUIVALENCE (ST2(2001),UK2(1))
      EQUIVALENCE (ST2(3801),EVECTS(1))
      EQUIVALENCE (ST2(1),DMAT(1))
      EQUIVALENCE (ST2(1001),DMAT(1))
C
C***** EQUIVALENCE FOR PHASE 4 (OBTAIN OUTPUT RESPONSES)
C
      EQUIVALENCE (ST2(2001),Y1(1))
      EQUIVALENCE (ST2([illegible]),Y2(1))
      EQUIVALENCE (ST1(S24),Y3(1))
      EQUIVALENCE (ST2(5151),U2(1))
      EQUIVALENCE (ST2 [illegible],U3(1))
      EQUIVALENCE (ST2([illegible]),FC2(1))
      EQUIVALENCE (ST2(5401),FC3(1))
      EQUIVALENCE (ST2(3321),UK2(1))
C
C***** EQUIVALENCE FOR PHASE 5 (OBTAIN COMPLETE OUTPUT SPECTRUM)
C
      EQUIVALENCE (ST1(1),FR(1))
      EQUIVALENCE (ST1(101),IPF(1))
      EQUIVALENCE (ST1([illegible]),Y(1))
      EQUIVALENCE (ST1(311),YLG(1))
      DATA C,L,E,IS/2H C,2H L,2H E,2H I/
      DATA R,G,VV,CV,CC,VC/2H R,2H G,2HVV,2HCV,2HCC,2HVC/
      DATA NC,NL,NR,ND/2HNC,2HNL,2HNR,2HND/
C
C******   MAX CIRCUIT CONFIGURATION IS 30 NODES AND 75 BRANCHES
C
      CALL SECOND (T0)
      NNODE=0
      MXNODE=30
      MXBR=75
      MXPRT=25
      MXSTV=20
C
C******   MAX NONLINEAR ELEMENTS  IS 10 (DEPENDENT TYPE LE 5)
C******   MAX INDEPENDENT SOURCES IS 2
C******   MAX TOTAL CAPACITORS AND INDUCTORS IS 20
C
      NCAP=0
      NRES=0
      NIND=0
      NCS=0
      NDUP=0
      NDCS=0
      NOUT=1
      NOUT=2
C
C******   READ TITLE CARD
```

```
C
      READ (5,212) (TITLE(J),J=1,80)
C
C*****    WRITE TITLE LINE AND PRINT HEADING OF NETWORK DESCRIPTION
C
      WRITE (6,214) (TITLE(J),J=1,80)
C
C***** READ,*DECODE*,AND WRITE DESIRED OPTIONS
C
      READ (5,...) (TITLE(I),I=1,10)
      CALL ... (TITLE,ED,FS,...,SE,NM,PR,PC,AP)
      WRITE (6,...)
      WRITE (6,...)
      WRITE (6,...)
      WRITE (6,...)
C
C*****     READ ANALYSIS PARAMETER CARD
C
      READ (5,...) NLELEM,...,NFREQ,LUNIT,INTYP
      ...
C
C***** READ LINEAR CIRCUIT DESCRIPTION
C
      DO 100 KK=1,NLELEM
         READ (5,...) NEG,N1,N2,NT,VAL,ICT,OUTPT,NCT
C
C***** FORM LINEAR ELEMENT TOPOLOGY ARRAYS
C
         K=NEG
         ...=K
         [illegible]
         [illegible]
         [illegible]
         [illegible]
         [illegible]
C
C***** IF PRESENT BRANCH IS AN OUTPUT,EXTRACT IT AS A PORT
C
         KEYV=...
         IF (...LE.0) GO TO 102
         CALL ... (...,TR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,KEYV,IS,0.0,
     1   ...)
         [illegible]
         [illegible]
         [illegible]
         ...=NOUT
         [illegible]
  102    ...(NFROM(K),NTO(K))
C
C***** SORT LINEAR ELEMENTS
C
         [illegible]
         [illegible]
         ...(K).EQ.6)) GO TO 112
         [illegible]
         [illegible]
         [illegible]
C
C***** [illegible]
C
```

```
          KEY(K)=8
          NDCS=NDCS+1
          GO TO 110
C
C***** V C V S
C
  104     KEY(K)=3
          NDVS=NDVS+1
          GO TO 110
C
C***** C C V S
C
  106     KEY(K)=4
          NDVS=NDVS+1
          GO TO 110
C
C***** V C C S
C
  108     KEY(K)=7
          NDCS=NDCS+1
C
C***** WRITE DEPENDENT SOURCE BRANCH INFORMATION
C
  110     WRITE (6,228) BR(K),NFROM(K),NTO(K),TYPE(K),VALUE(K),ICONT(K)
          GO TO 120
C
C***** RESISTIVE BRANCH
C
  112     KEY(K)=5
          NRES=NRES+1
          GO TO 118
C
C***** CAPACITIVE BRANCH
C
  114     NCAP=NCAP+1
          KEY(K)=2
          GO TO 118
C
C***** INDUCTIVE BRANCH
C
  116     NIND=NIND+1
          KEY(K)=6
C
C***** WRITE R,L,C BRANCH INFORMATION
C
  118     WRITE (6,230) BR(K),NFROM(K),NTO(K),TYPE(K),VALUE(K)
  120 CONTINUE
C
C*****READ NONLINEAR ELEMENT INFORMATION
C
      KK=0
      DO 128 K=1,NNELEM
          KK=KK+1
C
C***** READ NONLINEAR ELEMENT TOPOLOGY
C
          READ (5,232) NEG,N1,N2,NT,ICT,JCT
          IF (NT.EQ.2HTR) GO TO 126
C
C***** READ POLYNOMIAL COEFFICIENTS FOR THE NONLINEARITY
```

```
C
         IF (NT.EQ.ND) GO TO 122
         READ (5,234) (COFF(KK,I),I=1,3)
         GO TO 124
  122    READ (5,236) (COFF(KK,I),I=1,9)
  124    NLBN(KK)=NEG
         BR(NEG)=NEG
         NFROM(NEG)=N1
         NTO(NEG)=N2
         TYPE(NEG)=NT
         ICONT(NEG)=ICT
         JCONT(KK)=JCT
         GO TO 128
  126 CALL LTRANS (NEG,N1,NADD,KK,NLBN,BR,NFROM,NTO,TYPE,ICONT,VALUE,NNO
     1DE,KEY)
  128 CONTINUE
      NNELEM=KK
C
C***** WRITE NONLINEAR ELEMENT INFORMATION
C
      WRITE (6,252)
      WRITE (6,254)
      DO 132 I=1,NNELEM
         N=NLBN(I)
         IF (TYPE(N).EQ.ND) GO TO 130
         WRITE (6,256) NFROM(N),NTO(N),TYPE(N),COFF(I,1),COFF(I,2),COFF(
    1    I,3)
         GO TO 132
  130    WRITE (6,258) NFROM(N),NTO(N),TYPE(N),ICONT(N),JCONT(I),COFF(I,
    1    1),COFF(I,2),COFF(I,3)
         WRITE (6,260) COFF(I,4),COFF(I,5),COFF(I,6)
         WRITE (6,260) COFF(I,7),COFF(I,8),COFF(I,9)
  132 CONTINUE
C
C*****READ AND WRITE GENERATOR INFORMATION
C
      READ (5,238) N1,N2,NT,ZS
      IF ((NT.EQ.R).OR.(NT.EQ.G)) GO TO 134
      IF (NT.EQ.C) GO TO 136
      CALL CKTPRT (NADD,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,6,NT,ZS)
      NIND=NIND+1
      NZT=3
      GO TO 138
  134 CALL CKTPRT (NADD,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,5,NT,ZS)
      NRES=NRES+1
      NZT=1
      GO TO 138
  136 CALL CKTPRT (NADD,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,2,NT,ZS)
      NCAP=NCAP+1
      NZT=2
  138 CALL CKTPRT (NADD,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,9,IS,0.00)
      NCS=NCS+1
      NCONT(1)=1
      DO 140 I=1,NFREQ
  140 READ (5,239) AMP(I),PHASE(I),FREQ(I),HFR(I),NSTPS(I)
      WRITE (6,240)
      WRITE (6,242) N1,N2,ZS,NT
      WRITE (6,244) LUNIT
      DO 142 I=1,NFREQ
  142 WRITE (6,246) I,FREQ(I),AMP(I),PHASE(I)
```

```
      IF (FS.NE.1) GO TO 146
      MXINC=0
      DO 144 I=1,NFREQ
  144 MXINC=MAX0(MXINC,NSTPS(I))
      WRITE (6,208) INTYP,MXINC
  146 IF (MX.NE.1) GO TO 154
C
C***** READ AND WRITE SECOND-GENERATOR INFORMATION
C
      NFREQ=NFREQ+1
      READ (5,248) N1,N2,NT,ZS1,AMP(NFREQ),FREQ(NFREQ),PHASE(NFREQ)
      IF ((NT.EQ.R).OR.(NT.EQ.G)) GO TO 148
      IF (NT.EQ.C) GO TO 150
      CALL CXTPRT (NADD,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,6,NT,ZS1)
      NIND=NIND+1
      NZT1=3
      GO TO 152
  148 CALL CXTPRT (NADD,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,5,NT,ZS1)
      NRES=NRES+1
      NZT1=1
      GO TO 152
  150 CALL CXTPRT (NADD,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,2,NT,ZS1)
      NCAP=NCAP+1
      NZT1=2
  152 CALL CXTPRT (NADD,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,35,IS,0.0)
      NCS=NCS+1
      WRITE (6,242) N1,N2,ZS1,NT
      WRITE (6,244) LUNIT
      WRITE (6,246) NFREQ,FREQ(NFREQ),AMP(NFREQ),PHASE(NFREQ)
C
C*****FORM THE APPROPRIATE AUGMENTED LINEAR NETWORK
C
  154 NCT=NOUT
      DO 156 K=1,NNELEM
         N=NLBN(K)
         ICON=ICONT(N)
         JCON=JCONT(K)
         NNODE=MAX0(NNODE,NFROM(N),NTO(N))
         KEYV=KEYV+1
         NCT=NCT+1
         CALL CAGMNT (K,N,KEYV,NADD,ICON,JCON,NCT,BR,NFROM,NTO,TYPE,ICON
     1   T,VALUE,KEY)
  156 CONTINUE
      IF (MX.NE.1) GO TO 158
      LOSRC=NCT+1
      NCONT(LOSRC)=LOSRC
  158 NELEM=NADD
C
C*****     SORT ELEMENT DATA
C
      NICS=NCS
      CALL CSORT (NELEM,BR,NFROM,NTO,TYPE,ICONT,VALUE,KEY)
C
C***** COMBINE PORTS WHICH APPEAR ACROSS SAME NODE PAIR
C
      CALL CRDPRT (NELEM,BR,NFROM,NTO,KEY,TYPE,VALUE,ICONT)
C
C***** CONSECUTIVELY NUMBER THE EXTRACTED "INDEPENDENT" PORTS
C
      DO 166 I=1,NICS
```

```
      IF (NCONT(I).EQ.I) GO TO 166
      J=I+1
  160 IF (NCONT(J).GT.I) GO TO 162
      IF (J.EQ.NICS) GO TO 166
      J=J+1
      GO TO 160
  162 DO 164 K=J,NICS
  164 IF (NCONT(K).EQ.J) NCONT(K)=I
  166 CONTINUE
C
C***** RENUMBER THE CONTROLLING PORTS FOR THE NONLINEAR ELEMENTS
C***** AND ASSIGN NUMERICAL IDENTIFIER(JCONT( )) WITH EACH NONLINEAR
C***** ELEMENT TYPE
C
      DO 168 K=1,NICS
  168 NLBN(K)=NCONT(K)
      K1=NOUT
      J3=NOUT
      DO 180 K=1,NNELEM
      K1=K1+1
      J1=J3+1
      J2=J1+1
      J3=J2+1
      IF (NTYPE(K).EQ.NR) GO TO 174
      IF (NTYPE(K).EQ.NC) GO TO 170
      IF (NTYPE(K).EQ.NL) GO TO 172
      IF (NTYPE(K).EQ.ND) GO TO 178
  170 JCONT(K)=1
      GO TO 176
  172 JCONT(K)=2
      GO TO 176
  174 JCONT(K)=4
  176 NCDUM=NLBN(K1)
      NCONT(J1)=NCDUM
      NCONT(J2)=NCDUM
      NCONT(J3)=NCDUM
      GO TO 180
  178 NCONT(J1)=NLBN(K1)
      NCONT(J2)=NLBN(K1+1)
      K1=K1+2
      NCONT(J3)=NLBN(K1)
      JCONT(K)=3
  180 CONTINUE
      IF (NK.NE.1) GO TO 182
      LGERO=NOUT+NNELEM+NNELEM+NNELEM+1
      NCONT(LGERO)=NLBN(NICS)
  182 NSTV=NCAP+NIND
      NBR=NELEM
C
C*****PRINT AUGMENTED LINEAR NETWORK DESCRIPTION
C
      WRITE (6,294)
      WRITE (6,220)
      WRITE (6,222)
      DO 184 K=1,NBR
  184 WRITE (6,226) BR(K),NFROM(K),NTO(K),TYPE(K),VALUE(K),ICONT(K)
C
C****  WRITE EXTRACTED PORT INFORMATION
C
      WRITE (6,288)
```

```
      NDUM=NDR-NCS
      DO 183 I=1,NCS
         NPORT(I)=I
         NDUM=NDUM+1
         WRITE (6,270) NPORT(I),NFROM(NDUM),NTO(NDUM)
  183 CONTINUE
C
C*****     ZERO OUT A MATRIX
C
      DO 188 I=1,MMNODE
      DO 188 J=1,MMDR
  188 A(I,J)=0
C
C*****     STORE ENTRIES INTO A MATRIX
C
      DO 190 K=1,NELEM
         FROM=NFROM(K)
         TO=NTO(K)
         IF (FROM.NE.0) A(FROM,K)=1
         IF (TO.NE.0) A(TO,K)=-1
  190 CONTINUE
      IF (DB.NE.1) GO TO 194
C
C*****     PRINT THE A MATRIX FOR DEBUG RUN
C
      WRITE (6,272)
      DO 192 I=1,NNODE
  192 WRITE (6,274) (A(I,J),J=1,NDR)
C
C*****     FORMULATE HYBRID EQUATIONS
C
  194 CALL FHYBRD (NDR,NNODE,DB,NPORT1,ANSBOL,II,DR,TYPE,ICONT,VALUE,A,[illegible]
     1EADER,ANS,EXTR1,EXTR2,MMNODE,MMDR)
C
C*****     FORMULATE STATE EQUATIONS
C
      CALL STATE (NPORT1,ANSBOL,II,NDR,NSTV,SE,AMAT,BMAT,CMAT,DMAT,VALU[illegible]
     1E,ANS,MMDR,MMSTV,MMPRT)
C
C*****OBTAIN AND PRINT THE EIGENVALUES AND THE EIGENVECTORS OF THE
C*****MATRIX A
C
      CALL FEVEV (AMAT,NSTV,MMSTV,EVALS,EVECTS,VALUE,IERR)
C
C***** PRINT EIGENVALUE AND EIGENVECTOR INFORMATION, IF DESIRED
C
      IF (NI.NE.1) GO TO 200
      WRITE (6,278)
      DO 196 I=1,NSTV
  196 WRITE (6,279) EVALS(I)
      WRITE (6,280)
      DO 198 I=1,NSTV
  198 WRITE (6,282) (EVECTS(I,J),J=1,NSTV)
C
C
C*****DETERMINE AND PRINT THE INVERSE OF YSD AT THE EXTRACTED PORTS
C
  200 CALL [illegible] ([illegible],EVECTS,EVALS,[illegible],BMAT,CMAT,DMAT,[illegible],NPCT
     [illegible]
      CALL [illegible]
```

```
C
C***** START FREQUENCY SWEEP, IF DESIRED
C
      [illegible]
      IF [illegible]
C
C***** [illegible] VALUES FOR FREQUENCY INCREMENTS
C
      [illegible]
C
C***** [illegible] THE FIRST ORDER RESPONSE
C
  [illegible] EVALS,Y1,NSFREQ,DMAT,ZS,ZS1,LOSR
     [illegible]
C
C***** [illegible] THE SECOND ORDER RESPONSE
C
      [illegible] DLS,DHAT,CHAT,Y1,Y2,NSFREQ,N2,NID
     [illegible]
C
C***** [illegible] THE THIRD ORDER RESPONSE
C
      [illegible] DLS,DHAT,CHAT,Y1,Y2,NSFREQ,N3,Y3,
     [illegible]
C
C***** [illegible] TRANSFER FUNCTIONS AND RESPONSES
C
      [illegible],NP,NOUT,NWRT)
      [illegible],F2D,N2,NP,NOUT,NWRT)
      [illegible],F3D,N3,NP,NOUT,NWRT)
C
C***** [illegible] OUTPUT SPECTRUM BY COMBINING THE VARIOUS
C***** [illegible]
C
      [illegible]
      [illegible] NFREQ,NIPT,NSFRPT,FR,Y,NOUT,N2,N3,IPT,YLG,
     [illegible]
C
C***** [illegible] VALUES
C
  [illegible]
      [illegible]
C
  [illegible]
  [illegible]FREQUENCY SWEEP TYPE:,2X,A3,/1X,2OHMAXIMUM NUMBER [illegible]
  [illegible]
  [illegible] DESCRIPTION:,/1H ,10A4,))
```

```
  218 FORMAT (//,1X,15HLINEAR ELEMENTS)
  220 FORMAT (1H0,6HBRANCH,4X,4HFROM,3X,2HTO,5X,7HELEMENT,3X,7HELEMENT,3
     1X,7HCONTROL,/,1X,6HNUMBER,4X,4HNODE,2X,4HNODE,5X,4HTYPE,6X,5HVALUE
     2,4X,6HBRANCH)
  222 FORMAT (1H ,6(1H.),4X,4(1H.),2X,4(1H.),4X,7(1H.),3X,7(1H.),3X,7(1H
     1.))
  224 FORMAT (2I2,I1,2A3)
  226 FORMAT (2I3,A2,E10.3,I3,I1,A1)
  228 FORMAT (2X,I3,6X,I3,3X,I3,7X,A2,4X,E10.3,3X,I3)
  230 FORMAT (2X,I3,6X,I3,3X,I3,7X,A2,4X,E10.3)
  232 FORMAT (2I3,A2,2I3)
  234 FORMAT (2E10.3)
  236 FORMAT (8E10.3,/,E10.3)
  238 FORMAT (2I3,A2,E10.3)
  240 FORMAT (//,1X,19HSOURCE INFORMATION:)
  242 FORMAT (1H0,4HFROM,2X,I3,2X,2HTO,2X,I3,5X,9HIMPEDANCE,2X,E10.3,3X,
     1A2)
  244 FORMAT (1H0,9HFREQUENCY,5X,6HVALUE(,A3,1H),9X,9HAMPLITUDE,4X,10HPH
     1ASE(DEG),/1H ,9(1H.),5X,10(1H.),9X,9(1H.),4X,10(1H.))
  246 FORMAT (1H ,4X,I2,3X,E10.3,3X,E10.3,2X,E10.3)
  248 FORMAT (2I3,A2,4E10.3)
  250 FORMAT (4E10.3,I2)
  252 FORMAT (///,1X,18HNONLINEAR ELEMENTS)
  254 FORMAT (1H0,2X,4HFROM,5X,2HTO,6X,4HTYPE,4X,7HCONTROL,5X,22HPOLYNOM
     1IAL COEFFICIENTS,/1H ,2X,4HNODE,4X,4HNODE,11X,3H(1),5X,3H(2),/1H ,
     22X,4(1H.),4X,4(1H.),3X,4(1H.),2X,11(1H.),3X,23(1H.))
  256 FORMAT (1H0,3X,I2,6X,I2,7X,A2,16X,3HA1=,E12.4,3X,3HA2=,E12.4,3X,3H
     1A3=,E12.4)
  258 FORMAT (1H ,38X,4HA02=,E12.4,2X,4HA11=,E12.4,2X,4HA30=,E12.4)
  260 FORMAT (1H ,38X,4HA03=,E12.4,2X,4HA21=,E12.4,2X,4HA12=,E12.4)
  262 FORMAT (1H0,3X,I2,6X,I2,7X,A2,3X,I2,6X,I2,3X,4HA10=,E12.4,2X,4HA01
     1=,E12.4,2X,4HA20=,E12.4)
  264 FORMAT (////,1X,36HAUGMENTED LINEAR NETWORK DESCRIPTION)
  266 FORMAT (2X,I3,6X,I3,3X,I3,7X,A2,4X,E10.3,3X,I3,6X)
  268 FORMAT (1H0,17HPORT ASSIGNMENTS:,/1H0,4X,4HPORT,4X,9HNODE PAIR,/1H
     1 ,3X,6HNUMBER,3X,4HFROM,2X,2HTO,/1H ,3X,6(1H.),3X,4(1H.),1X,4(1H.)
     2)
  270 FORMAT (6X,I2,5X,I2,3X,I2)
  272 FORMAT (///,9H A MATRIX)
  274 FORMAT (1H0,40I3)
  276 FORMAT (/,12H EIGENVALUES)
  278 FORMAT (/,2(3X,E12.4))
  280 FORMAT (/,13H MODAL MATRIX)
  282 FORMAT (/,4(3X,E12.3,3X,E12.3))
  284 FORMAT (1H0,25HTIME FOR FORMING ZOC(SEC),F10.4)
  286 FORMAT (1H ,39HTIME FOR OBTAINING OUTPUT SPECTRUM(SEC),F10.4)
  288 FORMAT (1H ,25HTOTAL EXECUTION TIME(SEC),F10.4)
C
      END
      SUBROUTINE DOPTNS (TITLE,DB,FS,MX,SE,NM,PR,PC,AP)
C
C**********************************************************************
C     *
C*****     THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:
C     *         1. SET THE FLAG VARIABLES FOR THE USER REQUESTED OPTIONS,
C     *            AND ALSO PRINT THESE OPTIONS.
C     *
C*****     THIS SUB-PROGRAM'S GLOSSARY OF FORTRAN NAMES:
C     *         TITLE      : USER REQUESTED OPTIONS
C     *         DB         : FLAG VARIABLE FOR DEBUG RUN
```

```
C      *         FS          : FLAG VARIABLE FOR FREQUENCY SWEEP
C      *         MX          : FLAG VARIABLE FOR TWO INPUT SOURCES
C      *         SE          : FLAG VARIABLE FOR STATE EQUATION PRINT-OUT
C      *         NM          : FLAG VARIABLE FOR EIGENVALUE-EIGENVECTOR
C      *                       INFORMATION PRINT-OUT
C      *         PR          : FLAG VARIABLE FOR POLE-RESIDUE INFORMATION
C      *                       OF ZOC PRINT-OUT
C      *         PC          : FLAG VARIABLE FOR COMPLETE SPECTRUM PRINT
C      *                       AND PLOT
C      *         AP          : FLAG VARIABLE FOR "ALL" PORT PRINT-OUT
C      *
C***********************************************************************
C
      INTEGER TITLE(1),DD,FS,SE,PR,PC,AP,LANS(2)
      DATA LANS(1),LANS(2)/3HYES,3H NO/
C
C***** INITIALIZE OPTION CONTROLLING FLAG VARIABLES
C
      DD=2
      FS=2
      MX=2
      SE=2
      NM=2
      PR=2
      PC=2
      AP=2
C
C***** RESET FLAG VARIABLE VALUES FOR THE REQUESTED OPTIONS
C
      DO 130 I=1,8
         IDUM=TITLE(I)
         IF (IDUM.EQ.2H  ) GO TO 132
         IF (IDUM.EQ.2HDD) GO TO 102
         GO TO 104
  102    DD=1
         GO TO 130
  104    IF (IDUM.EQ.2HFS) GO TO 106
         GO TO 108
  106    FS=1
         GO TO 130
  108    IF (IDUM.EQ.2HMX) GO TO 110
         GO TO 112
  110    MX=1
         GO TO 130
  112    IF (IDUM.EQ.2HSE) GO TO 114
         GO TO 116
  114    SE=1
         GO TO 130
  116    IF (IDUM.EQ.2HNM) GO TO 118
         GO TO 120
  118    NM=1
         GO TO 130
  120    IF (IDUM.EQ.2HPR) GO TO 122
         GO TO 124
  122    PR=1
         GO TO 130
  124    IF (IDUM.EQ.2HPC) GO TO 126
         GO TO 128
  126    PC=1
         GO TO 130
```

```
128     IF (IDUM.EQ.2HAP) AP=1
  130 CONTINUE
C
C***** PRINT THE OPTIONS LIST
C
  132 WRITE (6,124)
      WRITE (6,136) LANS(DB)
      WRITE (6,138) LANS(FS)
      WRITE (6,140) LANS(MX)
      WRITE (6,142) LANS(SE)
      WRITE (6,144) LANS(MM)
      WRITE (6,146) LANS(PR)
      WRITE (6,148) LANS(PC)
      WRITE (6,150) LANS(AP)
      RETURN
C
  134 FORMAT (1H0,23HUSER REQUESTED OPTIONS:)
  136 FORMAT (1H ,2X,19HDEBUG PRINT-OUT:,1X,A3)
  138 FORMAT (1H ,2X,27HFREQUENCY SWEEP CAPABILITY:,1X,A3)
  140 FORMAT (1H ,2X,18HTWO-INPUT CIRCUIT:,1X,A3)
  142 FORMAT (1H ,2X,25HSTATE EQUATION PRINT-OUT:,1X,A3)
  144 FORMAT (1H ,2X,35HEIGENVALUES MODAL MATRIX PRINT-OUT:,1X,A3)
  146 FORMAT (1H ,2X,40HOPEN-CIRCUIT IMPEDANCE MATRIX PRINT-OUT:,1X,A3)
  148 FORMAT (1H ,2X,30HCOMPLETE OUTPUT SPECTRUM PLOT:,1X,A3)
  150 FORMAT (1H ,2X,27HALL EXTRACTED PORT OUTPUTS:,1X,A3)
C
      END
      SUBROUTINE CASMNT (K,N,NKEY,NADD,ICON,JCON,NCT,BR,NFROM,NTO,TYPE,I
     1CONT,VALUE,KEY)
C
C*********************************************************************
C      *                                                             *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:            *
C      *      1. FORM THE AUGMENTED LINEAR NETWORK BY LUMPING THE      *
C      *         LINEAR PART OF EACH NONLINEAR ELEMENT WITH THE       *
C      *         EXISTING LINEAR NETWORK.                             *
C      *                                                             *
C******  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINE:              *
C      *      1. CXTPRT                                               *
C      *                                                             *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                *
C      *      K          : NUMBER OF THE NONLINEAR ELEMENT            *
C      *      N          : USER SPECIFIED BRANCH NUMBER FOR THE K-TH  *
C      *                   NONLINEAR ELEMENT                         *
C      *      NKEY       : KEY VALUE FOR THE NONLINEAR ELEMENT PORT DRAN*
C      *      NADD       : CURRENT HIGHEST BRANCH NUMBER IN THE LINEAR *
C      *                   NETWORK                                   *
C      *      ICON       : NONLINEAR ELEMENT FIRST-CONTROLLING BRANCH NO*
C      *      JCON       : NONLINEAR ELEMENT SECOND-CONTROLLING BR NO. *
C      *      NCT        : CURRENT NONLINEAR ELEMENT PORT NUMBER      *
C      *      ARRAY NAMES AS DEFINED IN SUB-PROGRAM AMAIN             *
C      *                                                             *
C*********************************************************************
C
      INTEGER BR,TYPE,C,L,E,R,G,VV,VC,CV,CC
      DIMENSION BR(1), NFROM(1), NTO(1), TYPE(1), ICONT(1)
      DIMENSION VALUE(1)
      DIMENSION KEY(1)
      COMMON /001/ NTYPE(10),COEF(10,9)
      COMMON /012/ NCONT(32),JCONT(10)
```

```
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NDCS,NCS
      DATA C,L,G,VC,IS/2H C,2H L,2H G,2HVC,2H I/
      DATA NC,NL/2HNC,2HNL/
C
C*****APPLY A ZERO-VALUED CURRENT SOURCE ACROSS THE NONLINEAR ELEMENT
C
      NTYPE(K)=TYPE(N)
      TYPE(N)=IS
      VALUE(N)=0.00
      KEY(N)=NKEY
      NCONT(NCT)=NCT
      NCS=NCS+1
C
C*****CHECK FOR A DEPENDENT NONLINEAR ELEMENT
C
      IF (ICON.LE.0) GO TO 104
      NCT=NCT+1
      NKEY=NKEY+1
      CALL CXTFRT (NADD,DR,TYPE,VALUE,NFROM,NTO,KEY,NFROM(ICON),NTO(ICON
     1),NKEY,IS,0.00)
      NCS=NCS+1
      VALUE(NADD)=0.000
      KEY(NADD)=NKEY
      NCONT(NCT)=NCT
      NKEY=NKEY+1
      NCT=NCT+1
      CALL CXTFRT (NADD,DR,TYPE,VALUE,NFROM,NTO,KEY,NFROM(JCON),NTO(JCON
     1),NKEY,IS,0.00)
      NCS=NCS+1
      NCONT(NCT)=NCT
C
C*****COMBINE THE LINEAR PART OF THE NONLINEARITY WITH THE LINEAR NTWK
C
      IF (COFF(K,2).EQ.0.00) GO TO 102
      CALL CXTFRT (NADD,DR,TYPE,VALUE,NFROM,NTO,KEY,NFROM(N),NTO(N),8,VC
     1,COFF(K,2))
      ICONT(NADD)=JCONT(K)
      JCONT(K)=0
      NDCS=NDCS+1
  102 IF (COFF(K,1).EQ.0.00) RETURN
      CALL CXTFRT (NADD,DR,TYPE,VALUE,NFROM,NTO,KEY,NFROM(N),NTO(N),8,VC
     1,COFF(K,1))
      ICONT(NADD)=ICONT(N)
      ICONT(N)=0
      NDCS=NDCS+1
      RETURN
C
C*****INDEPENDENT TYPE NONLINEARITY
C
  104 IF (COFF(K,1).EQ.0.00) RETURN
      IF (NTYPE(K).EQ.NC) GO TO 105
      IF (NTYPE(K).EQ.NL) GO TO 103
      CALL CXTFRT (NADD,DR,TYPE,VALUE,NFROM,NTO,KEY,NFROM(N),NTO(N),5,G,
     1COFF(K,1))
      NRES=NRES+1
      RETURN
  105 CALL CXTFRT (NADD,DR,TYPE,VALUE,NFROM,NTO,KEY,NFROM(N),NTO(N),2,C,
     1COFF(K,1))
      NCAP=NCAP+1
      RETURN
```

```
  108 RECVAL=1.0000/COFF(K,1)
      CALL CMTPRT (NADD,DR,TYPE,VALUE,NFROM,NTO,KEY,NFROM(N),NTO(N),G,L,
     1RECVAL)
      NIND=NIND+1
      RETURN
C
      END
      SUBROUTINE CROPRT (NDR,DR,NFROM,NTO,KEY,TYPE,VALUE,ICONT)
C
C*********************************************************************
C     *                                                              *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTIONS:          *
C     *        1. IDENTIFY ALL PARALLEL CAPACITOR,INDUCTOR,OR CURRENT *
C     *           SOURCE BRANCHES.                                    *
C     *        2. COMBINE THESE PARALLEL BRANCHES.                    *
C     *                                                              *
C******  THIS SUB-PROGRAM'S GLOSSARY OF FORTRAN NAMES:                *
C     *        NDR        : TOTAL NUMBER OF LINEAR ELEMENT BRANCHES   *
C     *        ALL OTHER VARIABLE AND ARRAY NAMES AS DEFINED PREVIOUSLY *
C     *        IN SUB-PROGRAMS AMAIN AND CASMAT.                      *
C     *                                                              *
C*********************************************************************
C
      INTEGER DR(1),NFROM(1),NTO(1),KEY(1),TYPE(1),ICONT(1)
      DIMENSION VALUE(1)
      COMMON /01S/ NCONT(22),JCONT(10)
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NDCS,NCS
      ICOMB=0
      IFLAG=0
      NONSRC=NCAP+NIND+NRES+NDVS+NDCS
C
C***** IDENTIFY PARALLEL CAPACITOR, INDUCTOR, AND CURRENT SOURCE BRANCHES
C
  102 IFLAG=IFLAG+1
      GO TO (104,106,108), IFLAG
  104 N=NCAP
      NF=0
      KS=1
      GO TO 110
  106 N=NIND+NRES+NDVS
      KS=NF+NDVS+NRES+1
      GO TO 110
  108 N=NCS+NDCS
      KS=NF+NDCS+1
  110 IF (N.EQ.0) GO TO 102
      NF=NF+N
      I=KS
  112 IF (I.GT.NF) GO TO 126
      IF (DR(I).EQ.0) GO TO 124
      I1=I+1
      DO 122 J=I1,NF
         IF ((NFROM(I).EQ.NFROM(J)).AND.(NTO(I).EQ.NTO(J)).AND.(TYPE(I).
     1   EQ.TYPE(J))) GO TO 114
         GO TO 122
  114    ICOMB=ICOMB+1
         DR(J)=0
         GO TO (116,118,120), IFLAG
C
C***** PARALLEL CAPACITOR
C
```

```
  116    VALUE(I)=VALUE(I)+VALUE(J)
         NCAP=NCAP-1
         GO TO 122
C
C***** PARALLEL INDUCTOR
C
  118    VALUE(I)=VALUE(I)*VALUE(J)/(VALUE(I)+VALUE(J))
         NIND=NIND-1
         GO TO 122
  120    ITN=I-NONERO
         JTN=J-NONERO
         NCONT(JTN)=NCONT(ITN)
         NCS=NCS-1
  122 CONTINUE
  124 I=I+1
      GO TO 112
  126 IF (IFLAG.LT.3) GO TO 102
      IF (ICOND.EQ.0) RETURN
C
C***** REMOVE ALL PARALLEL BRANCHES IDENTIFIED PREVIOUSLY
C
      I=1
  128 IF (I.GT.NDR) GO TO 136
      IF (DR(I).EQ.0) GO TO 130
      GO TO 134
  130 I1=NDR-1
      DO 132 J=I,I1
         DR(J)=DR(J+1)
         NFROM(J)=NFROM(J+1)
         NTO(J)=NTO(J+1)
         KEY(J)=KEY(J+1)
         VALUE(J)=VALUE(J+1)
         TYPE(J)=TYPE(J+1)
         ICONT(J)=ICONT(J+1)
  132 CONTINUE
      NDR=NDR-1
      GO TO 128
  134 I=I+1
      GO TO 128
  136 RETURN
C
      END
      SUBROUTINE CSORT (NELEM,DR,NFROM,NTO,TYPE,ICONT,VALUE,KEY)
C
C*********************************************************************
C      *                                                             *
C*****   THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:           *
C      *        1. ARRANGE THE BRANCHES OF THE AUGMENTED LINEAR NETWORK *
C      *           IN AN ORDER SUITABLE FOR HYBRID AND STATE EQUATION *
C      *           FORMULATION.                                      *
C      *                                                             *
C*****   THIS SUB-PROGRAM'S GLOSSARY OF FORTRAN NAMES:               *
C      *        NELEM    : TOTAL NUMBER OF LINEAR ELEMENT BRANCHES   *
C      *        ARRAY NAMES AS DEFINED IN SUB-PROGRAM AMAIN          *
C      *                                                             *
C*********************************************************************
C
C*****    REARRANGE ELEMENT DATA TO ORDERING REQUIRED FOR HYBRID AND
C*****    STATE SUBROUTINES VIA SHELL SORT METHOD
C
```

```
      INTEGER BR,TYPE
      DIMENSION BR(1), NFROM(1), NTO(1), TYPE(1), ICONT(1)
      DIMENSION VALUE(1)
      DIMENSION KEY(1)
      COMMON /018/ NCONT(32),JCONT(10)
      J=1
  102 J=2*J
      I=NELEM/J
      IF (I.EQ.0) GO TO 108
      L=1
      M=I+1
  104 IF (M.GT.NELEM) GO TO 102
      LL=L
      MM=M
      IF (KEY(M).GE.KEY(L)) GO TO 106
      ITEMP=KEY(M)
      KEY(M)=KEY(L)
      KEY(L)=ITEMP
      TEMP=VALUE(M)
      VALUE(M)=VALUE(L)
      VALUE(L)=TEMP
      ITEMP=BR(M)
      BR(M)=BR(L)
      BR(L)=ITEMP
      ITEMP=NFROM(M)
      NFROM(M)=NFROM(L)
      NFROM(L)=ITEMP
      ITEMP=NTO(M)
      NTO(M)=NTO(L)
      NTO(L)=ITEMP
      ITEMP=TYPE(M)
      TYPE(M)=TYPE(L)
      TYPE(L)=ITEMP
      ITEMP=ICONT(M)
      ICONT(M)=ICONT(L)
      ICONT(L)=ITEMP
      L=L-I
      IF (L.LT.1) GO TO 106
      M=M-I
      GO TO 104
  106 L=LL+1
      M=MM+1
      GO TO 104
  108 RETURN
C
      END
      SUBROUTINE CXTPRT (J,BR,TYPE,VALUE,NFROM,NTO,KEY,N1,N2,KEYU,NT,T)
C
C***********************************************************************
C       *                                                              *
C*******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:            *
C       *       1. ADD A BRANCH TO THE LINEAR NETWORK.                 *
C       *                                                              *
C*******  THIS SUB-PROGRAM'S GLOSSARY OF FORTRAN NAMES:                *
C       *       J            : NEW BRANCH NUMBER                       *
C       *       NT           : NEW BRANCH'S ELEMENT TYPE               *
C       *       T            : NEW BRANCH'S ELEMENT VALUE              *
C       *       N1           : NEW BRANCH'S  FROM (+) NODE NUMBER      *
C       *       N2           : NEW BRANCH'S  TO (-) NODE NUMBER        *
C       *       KEYU         : NEW BRANCH'S KEY VALUE                  *
```

```
C       *        ARRAY NAMES AS DEFINED IN SUB-PROGRAM AMAIN            *CPT
C       *                                                               *CPT
C***********************************************************************CPT
C                                                                        CPT
      INTEGER BR,TYPE,C,L,E,IS,R,G,VV,CV,CC,VC                           CPT
      DIMENSION BR(1), NFROM(1), NTO(1), TYPE(1), VALUE(1), KEY(1)       CPT
C                                                                        CPT
C***** ADD A BRANCH                                                      CPT
C                                                                        CPT
      J=J+1                                                              CPT
      BR(J)=J                                                            CPT
      TYPE(J)=NT                                                         CPT
      VALUE(J)=T                                                         CPT
      NFROM(J)=N1                                                        CPT
      NTO(J)=N2                                                          CPT
      KEY(J)=KEYV                                                        CPT
      RETURN                                                             CPT
C                                                                        CPT
      END                                                                CPT
```

```
      SUBROUTINE DFTREE (NROW,NCOL,INDCOL,A,MV)                         DTR
C                                                                       DTR
C***********************************************************************DTR
C     *                                                                *DTR
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:             *DTR
C     *          1. FIND THE PROPER TREE FROM THE INCIDENCE MATRIX.     *DTR
C     *                                                                *DTR
C******  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINES:              *DTR
C     *          1. DIAECH                                             *DTR
C     *                                                                *DTR
C******  THIS SUB-PROGRAM≠S GLOSSARY OF FORTRAN NAMES:                 *DTR
C     *         NROW       : NUMBER OF ROWS IN THE INCIDENCE (A) MATRIX *DTR
C     *         NCOL       : NUMBER OF COLUMNS IN THE INCIDENCE (A) MATRIX*DTR
C     *         INDCOL     : INDEPENDENT COLUMNS OF THE A MATRIX        *DTR
C     *                                                                *DTR
C***********************************************************************DTR
C                                                                       DTR
C         SUBROUTINE DFTREE TAKES THE MATRIX A, APPLIES SUBROUTINE DIAECDTR
C         AND FINDS THE INDEPENDENT COLUMNS IN A CLOSEST TO THE LEFT.   DTR
C         THESE INDEPENDENT COLUMNS MAKE UP THE TREE BRANCHES.          DTR
C                                                                       DTR
      INTEGER A,INDCOL(NROW),COL,TEMP                                   DTR
      DIMENSION A(MV,1)                                                 DTR
      L=1                                                               DTR
      TEMP=1                                                            DTR
      CALL DIAECH (NROW,NCOL,A,MV)                                      DTR
C                                                                       DTR
C         STEP THROUGH ROWS                                             DTR
C                                                                       DTR
      DO 104 K=1,NROW                                                   DTR
C                                                                       DTR
C         STEP THROUGH COLUMNS                                          DTR
C                                                                       DTR
        DO 102 J=TEMP,NCOL                                              DTR
C                                                                       DTR
C         FIND INDEPENDENT COLUMNS                                      DTR
C                                                                       DTR
C         TEST IF ELEMENT EQUAL TO ONE                                  DTR
C                                                                       DTR
          IF (A(K,J).NE.1) GO TO 102                                    DTR
C                                                                       DTR
C         RECORD INDEPENDENT COLUMN NUMBER                              DTR
C                                                                       DTR
          INDCOL(L)=J                                                   DTR
          L=L+1                                                         DTR
          TEMP=J+1                                                      DTR
          GO TO 104                                                     DTR
  102   CONTINUE                                                        DTR
  104 CONTINUE                                                          DTR
      RETURN                                                            DTR
C                                                                       DTR
      END                                                               DTR
      SUBROUTINE DHYBRD (NBR,NNODE,DEBUG,NPORT1,ANSCOL,II,BR,TYPE,ICONT,DHD
     1VALUE,A,HEADER,ANS,F3,F6,MV,ME)                                   DHD
C                                                                       DHD
C***********************************************************************DHD
C     *                                                                *DHD
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTIONS:            *DHD
C     *          1. PERFORM HYBRID ANALYSIS OF THE AUGMENTED LINEAR CKT. *DHD
C     *          2. ACT AS THE EXECUTIVE CALLING PROGRAM FOR PERFORMING *DHD
```

```
C      *          THE HYBRID ANALYSIS.                                  *
C      *                                                                *
C******  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINES:               *
C      *       1. DFTREE                                                *
C      *       2. DIAECH                                                *
C      *       3. DPRINT                                                *
C      *       4. DRAECH                                                *
C      *       5. DPRNT1                                                *
C      *                                                                *
C******  THIS SUB-PROGRAM≠S GLOSSARY OF FORTRAN NAMES:                  *
C      *       NBR       : NUMBER OF AUGMENTED LINEAR NETWORK BRANCHES  *
C      *       NNODE     : NUMBER OF INDEPENDENT NODES                  *
C      *       DEBUG     : DEBUG OPTION FLAG VARIABLE                   *
C      *       NPORT1    : ADDRESS FOR LOCATING FIRST COLUMN OF MATRIX A*
C      *                   IN THE HYBRID MATRIX                         *
C      *       ANSCOL    : ADDRESS FOR LOCATING FIRST COLUMN OF MATRIX B*
C      *       II        : ADDRESS FOR LOCATING FIRST ROW OF MATRIX A   *
C      *       ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN SUB-   *
C      *       PROGRAM AMAIN                                            *
C      *                                                                *
C*****************************************************************************
C
      INTEGER A,TYPE,ICONT,VH,CH,DCOL(75),ICOUNT(2),COUN,BEGIN,TEMP,ST,T
     1N,TP,PORT,HEADER,BR,RBR(75),ANSROW,ANSCOL,DEBUG,ISTP
      INTEGER R,G,C,E,CV,VV,CC,VC
      DIMENSION BR(1), TYPE(1), ICONT(1)
      DIMENSION VALUE(1)
      DIMENSION A(MV,1), HEADER(300)
      DIMENSION ANS(ME,1)
      DIMENSION F3(ME,1), F6(ME,1)
      COMMON /ETYPE/ R,G,L,C,E,IS,CV,VV,CC,VC
      DATA CH,VH/1HI,1HV/
      DO 102 I=1,NBR
         DCOL(I)=0
  102 RBR(I)=0
C
C         DETERMINE ELEMENTS MAKING UP THE TREE
C
      CALL DFTREE (NNODE,NBR,DCOL,A,MV)
C
C         REORDER A MATRIX INTO FOUR CLASSES
C
C              1.   TREE PORT BRANCHES (TP)
C              2.   TREE NON-PORT BRANCHES (TN)
C              3.   LINK NON-PORT BRANCHES (LN)
C              4.   LINK PORT BRANCHES (LP)
C
C         DCOL CONTAINS ORDERING OF A WITH TREE BRANCHES IN LEFTMOST
C         COLUMNS
C
      JJ=NNODE+1
      N=1
      DO 106 J=1,NNODE
         M=DCOL(J)
         DO 104 K=N,M
            IF (M.EQ.K) GO TO 106
            DCOL(JJ)=K
            [illegible]
      [illegible]
      [illegible]
```

```
      DO 108 I=N,NBR
  108 DCOL(I)=I
C
C          REORDER DCOL INTO FOUR CLASSES
C          ICOUNT(1) MARKS LAST PORT COLUMN OF TREE BRANCHES
C          ICOUNT(2) MARKS LAST NON-PORT COLUMN OF LINK BRANCHES
C
      ICOUNT(1)=1
      IT2=NNODE
      I=1
  110 DO 112 M=1,IT2
         MM=(NBR+1)*(I-1)+((3-(2*I))*M)
         ITEM=DCOL(MM)
         IF (TYPE(ITEM).NE.E.AND.TYPE(ITEM).NE.C.AND.TYPE(ITEM).NE.L.AND
     1   .TYPE(ITEM).NE.IS) GO TO 112
         ITEM1=ICOUNT(I)
         DCOL(MM)=DCOL(ITEM1)
         DCOL(ITEM1)=ITEM
         ICOUNT(I)=ICOUNT(I)+1-((I-1)*2)
  112 CONTINUE
      IF (I.EQ.2) GO TO 114
      ICOUNT(1)=ICOUNT(1)-1
      ICOUNT(2)=NBR
      IT2=NBR-NNODE
      I=2
      GO TO 110
C
C          REORDER THE A MATRIX AND THE ORIGINAL LABEL VECTOR TO
C          CORRESPOND TO THE REORDERED DCOL
C
  114 NN=2
      N=1
      BEGIN=1
      COUN=0
  116 ITEM=DCOL(N)
      IF (ITEM.EQ.BEGIN) GO TO 120
      ITEMP=BR(N)
      BR(N)=BR(ITEM)
      BR(ITEM)=ITEMP
      DO 118 J=1,NNODE
         TEMP=A(J,N)
         A(J,N)=A(J,ITEM)
  118 A(J,ITEM)=TEMP
      COUN=COUN+1
      DCOL(N)=-DCOL(N)
      N=ITEM
      GO TO 116
  120 DCOL(N)=-DCOL(N)
      IF (COUN.EQ.(NBR-1)) GO TO 126
      DO 124 I=NN,NBR
         ITEM=DCOL(I)
         IF (ITEM.EQ.I) GO TO 122
         IF (ITEM.LT.0) GO TO 124
         BEGIN=I
         N=I
         GO TO 116
  122    COUN=COUN+1
         DCOL(I)=-DCOL(I)
         NN=I
  124 CONTINUE
```

```
  126 DO 128 N=1,NBR
  128 BCOL(N)=IABS(BCOL(N))
C
C         REDUCE REORDERED A MATRIX TO ROW ECHELON FORM
C
      CALL DIRECH (NNODE,NBR,A,MU)
C
C         BACK SUBSTITUTE A MATRIX
C
      DO 130 I=2,NNODE
         LROW=I-1
      DO 130 J=1,LROW
         IFCOL=I
         ITEMP=A(J,IFCOL)
      DO 130 K=I,NBR
  130 A(J,K)=A(J,K)-A(I,K)*ITEMP
C
C         FORMULATE THE ELEMENT CHARACTERISTICS
C
C         TP IS THE NUMBER OF COLUMNS IN F1 AND F5
C         TN IS THE NUMBER OF COLUMNS IN F2 AND F6
C         LN IS THE NUMBER OF COLUMNS IN F3 AND F7
C         LP IS THE NUMBER OF COLUMNS IN F4 AND F8
C
      TP=ICOUNT(1)
      TN=NNODE-ICOUNT(1)
      LN=ICOUNT(2)-NNODE
      LP=NBR-ICOUNT(2)
      PORT=TP+LP
      NPORT=TN+LN
      ANSROW=NBR
      ANSCOL=NBR+PORT
      WRITE (6,272)
      IF (TP.EQ.0) GO TO 132
      IF (DEBUG.NE.1) GO TO 132
      WRITE (6,274) (BR(I),I=1,TP)
  132 J=TP+1
      IF (TN.EQ.0) GO TO 134
      IF (DEBUG.NE.1) GO TO 134
      WRITE (6,276) (BR(I),I=J,NNODE)
  134 J=NNODE+1
      JJ=NNODE+LN
      IF (LN.EQ.0) GO TO 136
      IF (DEBUG.NE.1) GO TO 136
      WRITE (6,278) (BR(I),I=J,JJ)
  136 J=JJ+1
      IF (LP.EQ.0) GO TO 138
      IF (DEBUG.NE.1) GO TO 138
      WRITE (6,280) (BR(I),I=J,NBR)
C
C         ZERO ANS MATRIX
C
  138 DO 140 I=1,ANSROW
      DO 140 J=1,ANSCOL
  140 ANS(I,J)=0.0
      DO 144 I=1,NPORT
         DO 142 J=1,TN
  142    F6(I,J)=0.0
      DO 144 J=1,LN
  144 F4(I,J)=0.0
```

```
      KOUNT=ICOUNT(1)
      K=0
      J=1
      DO 146 I=1,NBR
         ITEM=BR(I)
  146 RBR(ITEM)=I
      IF (DEBUG.NE.1) GO TO 148
      WRITE (6,222) TP,TN,LN,LP
      WRITE (6,284) (BR(I),I=1,NBR)
  148 KOUNT=KOUNT+1
      MM=DCOL(KOUNT)
      ITEMP=ICONT(MM)
      ITEMP=RBR(ITEMP)
      IT1=PORT+J
      IF (TYPE(MM).EQ.G.OR.TYPE(MM).EQ.VC.OR.TYPE(MM).EQ.CC) GO TO 152
C
C          VOLTAGE SOURCE TYPE
C
      IF (KOUNT.GT.NNODE) GO TO 150
C
C          F2
C
      IT2=LN+J
      ANS(IT1,IT2)=1.
      IF (TYPE(MM).EQ.CV) GO TO 158
      IF (TYPE(MM).EQ.VV) GO TO 166
      F6(J,J)=-VALUE(MM)
      GO TO 156
  150 K=K+1
      F3(J,K)=1.
      IF (TYPE(MM).EQ.CV) GO TO 158
      IF (TYPE(MM).EQ.VV) GO TO 166
C
C          F7
C
      ANS(IT1,K)=-VALUE(MM)
      GO TO 156
C
C          CURRENT SOURCE TYPE
C
  152 IF (KOUNT.GT.NNODE) GO TO 154
      F6(J,J)=1.
      IF (TYPE(MM).EQ.VC) GO TO 166
      IF (TYPE(MM).EQ.CC) GO TO 158
C
C          F2
C
      IT2=LN+J
      ANS(IT1,IT2)=-VALUE(MM)
      GO TO 156
  154 K=K+1
C
C          F7
C
      ANS(IT1,K)=1.
      IF (TYPE(MM).EQ.VC) GO TO 166
      IF (TYPE(MM).EQ.CC) GO TO 158
      F3(J,K)=-VALUE(MM)
  156 J=J+1
      IF (KOUNT.NE.ICOUNT(2)) GO TO 148
```

```
      GO TO 174
C
C          CURRENT CONTROLLED
C
  158 IF (ITEMP.GT.TP) GO TO 160
C
C          F5
C
      IT2=NPORT+ITEMP
      ANS(IT1,IT2)=-VALUE(MM)
      GO TO 156
  160 IF (ITEMP.GT.NNODE) GO TO 162
      IT=ITEMP-TP
      F6(J,IT)=-VALUE(MM)
      GO TO 156
  162 IF (ITEMP.GT.ICOUNT(2)) GO TO 164
      IT=ITEMP-NNODE
C
C          F7
C
      ANS(IT1,IT)=-VALUE(MM)
      GO TO 156
  164 IT=ITEMP-ICOUNT(2)
C
C          F8
C
      IT2=NBR+TP+IT
      ANS(IT1,IT2)=-VALUE(MM)
      GO TO 156
C
C          VOLTAGE CONTROLLED
C
  166 IF (ITEMP.GT.TP) GO TO 168
C
C          F1
C
      IT2=NBR+ITEMP
      ANS(IT1,IT2)=-VALUE(MM)
      GO TO 156
  168 IF (ITEMP.GT.NNODE) GO TO 170
      IT=ITEMP-TP
C
C          F2
C
      IT2=LN+IT
      ANS(IT1,IT2)=-VALUE(MM)
      GO TO 156
  170 IF (ITEMP.GT.ICOUNT(2)) GO TO 172
      IT=ITEMP-NNODE
      F3(J,IT)=-VALUE(MM)
      GO TO 156
  172 IT=ITEMP-ICOUNT(2)
C
C          F4
C
      IT2=NPORT+TP+IT
      ANS(IT1,IT2)=-VALUE(MM)
      GO TO 156
  174 IF (BEDUG.NE.1) GO TO 196
      IF (LN.EQ.0) GO TO 184
```

```
C
C         WRITE F3 FOR DEBUG RUN
C
      WRITE (6,286)
      IT1=1
  176 IT2=LN
      IF ((IT2-IT1).GT.10) GO TO 180
      IF (IT2.EQ.IT1) GO TO 184
      WRITE (6,288)
      DO 178 I=1,NPORT
  178 WRITE (6,290) (F3(I,J),J=IT1,IT2)
      GO TO 184
  180 IT2=IT1+9
      WRITE (6,288)
      DO 182 I=1,NPORT
  182 WRITE (6,290) (F3(I,J),J=IT1,IT2)
      IT1=IT2+1
      GO TO 176
  184 IF (TP.EQ.0) GO TO 194
C
C         WRITE F6 FOR DEBUG RUN
C
      WRITE (6,292)
      IT1=1
  186 IT2=TN
      IF ((IT2-IT1).GT.10) GO TO 190
      IF (IT2.EQ.IT1) GO TO 194
      WRITE (6,288)
      DO 188 I=1,NPORT
  188 WRITE (6,290) (F6(I,J),J=IT1,IT2)
      GO TO 194
  190 IT2=IT1+9
      WRITE (6,288)
      DO 192 I=1,NPORT
  192 WRITE (6,290) (F6(I,J),J=IT1,IT2)
      IT1=IT2+1
      GO TO 186
  194 WRITE (6,294)
      CALL DPRINT (ANSCOL,ANSROW,ANS,ME)
C
C         ZERO OUT F6
C
  196 IF (TN.EQ.0) GO TO 206
      DO 204 J=1,TN
        KK=TP+J
      DO 204 I=1,NPORT
        IT1=PORT+I
        IF (LN.EQ.0) GO TO 200
C
C         CHANGE F7
C
        DO 198 K=1,LN
          LK=NNODE+K
  198   ANS(IT1,K)=ANS(IT1,K)-(F6(I,J)*FLOAT(A(KK,LK)))
  200   IF (LP.EQ.0) GO TO 204
C
C         CHANGE F8
C
        DO 202 K=1,LP
          LK=ICOUNT(2)+K
```

```
         IT2=NDR+TP+K
 202    ANS(IT1,IT2)=ANS(IT1,IT2)-(F6(I,J)*FLOAT(A(KK,LK)))
 204 CONTINUE
C
C          ZERO OUT F3
C
 206 IF (LN.EQ.0) GO TO 216
     DO 214 J=1,LN
        LK=NNODE+J
     DO 214 I=1,NPORT
        IT1=PORT+I
        IF (TN.EQ.0) GO TO 210
C
C          CHANGE F2
C
        DO 208 K=1,TN
           KK=TP+K
           IT2=LN+K
 208    ANS(IT1,IT2)=ANS(IT1,IT2)-(F3(I,J)*FLOAT(-A(KK,LK)))
 210    IF (TP.EQ.0) GO TO 214
C
C          CHANGE F1
C
        DO 212 K=1,TP
           IT2=NDR+K
 212    ANS(IT1,IT2)=ANS(IT1,IT2)-(F3(I,J)*FLOAT(-A(K,LK)))
 214 CONTINUE
C
C          FILL ANS MATRIX
C
 216 IF (DEBUG.NE.1) GO TO 218
     WRITE (6,293)
     CALL DPRINT (ANSCOL,ANSROW,ANS,ME)
 218 IF (LN.EQ.0.OR.TP.EQ.0) GO TO 222
C
C          STORE D1
C
     DO 220 I=1,TP
     DO 220 J=1,LN
        K=NNODE+J
 220 ANS(I,J)=A(I,K)
 222 LC=LN+1
     ITEMP=TP+1
     IF (ITEMP.GT.PORT.OR.LC.GT.NPORT) GO TO 226
C
C          STORE -D4 TRANSPOSE
C
     DO 224 I=ITEMP,PORT
        JJ=LC+I-ITEMP+NNODE
     DO 224 J=LC,NPORT
        II=J+1-LC+TP
 224 ANS(I,J)=-A(II,JJ)
 226 IF (TP.EQ.0) GO TO 230
C
C          STORE UNIT MATRIX ABOVE F5
C
     DO 228 I=1,TP
        LD=NPORT+I
 228 ANS(I,LD)=1.0
 230 IF (LP.EQ.0) GO TO 234
```

```
C
C         STORE UNIT MATRIX ABOVE F4
C
      II=TP+1
      DO 232 I=II,PORT
         LD=NPORT+I
  232 ANS(I,LD)=1.0
  234 ITEMP=TP+1
      LF=LD+TP
      LE=LD+1
      IF (ITEMP.GT.PORT.OR.LE.GT.LF) GO TO 238
C
C         STORE -D2 TRANSPOSE
C
      DO 236 I=ITEMP,PORT
         JJ=I-ITEMP+ICOUNT(2)+1
      DO 236 J=LE,LF
         II=J+1-LE
  236 ANS(I,J)=-A(II,JJ)
  238 LE=LF+LP
      LD=LF+1
      IF (TP.EQ.0.OR.LD.GT.LE) GO TO 242
C
C         STORE D2
C
      DO 240 I=1,TP
      DO 240 J=LD,LE
         K=ICOUNT(2)+1+J-LD
  240 ANS(I,J)=A(I,K)
  242 IF (DEBUG.NE.1) GO TO 244
      WRITE (6,298)
      CALL DPRINT (ANSCOL,ANSROW,ANS,ME)
C
C         REDUCE ANS MATRIX TO ECHELON FORM
C
  244 CALL DRAECH (NBR,ANSCOL,ANSCOL,1,1,ANS,MV,ME)
      ZERO=1.0000E-15
      IF (DEBUG.NE.1) GO TO 246
      WRITE (6,300)
      CALL DPRINT (ANSCOL,ANSROW,ANS,ME)
  246 DO 248 I=1,NBR
      DO 248 J=1,NPORT
         II=NBR+1-I
         IF (ABS(ANS(II,J)).LE.ZERO) ANS(II,J)=0.0
         IF (ANS(II,J).NE.0.) GO TO 250
  248 CONTINUE
  250 II=II+1
C
C         FILL COLUMN HEADING VECTOR FOR FINAL DPRINT OUT
C
      J=0
      IF (TP.EQ.0) GO TO 254
      DO 252 I=1,TP
         IT=2*I
         HEADER(IT)=BR(I)
         HEADER(IT-1)=CH
         I2=2*(PORT+I)
         HEADER(I2)=BR(I)
  252 HEADER(I2-1)=VH
  254 IF (LP.EQ.0) GO TO 258
```

```
      J=TP
      DO 256 I=1,LP
         J=J+1
         K=I+ICOUNT(2)
         IT=2*J
         HEADER(IT)=BR(K)
         HEADER(IT-1)=VH
         I2=2*(PORT+TP+I)
         HEADER(I2)=BR(K)
  256 HEADER(I2-1)=CH
  258 IT=4*PORT
      NPORT1=NPORT+1
      DO 260 I=II,NBR
      DO 260 J=NPORT1,ANSCOL
  260 IF (ABS(ANS(I,J)).LE.ZERO) ANS(I,J)=0.0
      IF (DEBUG.NE.1) GO TO 262
C
C           DPRINT FINAL ANS MATRIX FOR DEBUG RUN
C
      CALL DPRNT1 (IT,NPORT1,ANSCOL,II,NBR,HEADER,ANS,ME)
  262 IF (II.EQ.NBR) GO TO 266
C
C           BACK SUBSTITUTE FINAL ANSWER MATRIX
C
      IT1=ANSROW-II+1
      IT2=II+1
      DO 264 I=IT2,ANSROW
C
C           ANS(IRW,ICL) IS PIVOT ELEMENT USED TO ZERO ELEMENTS ABOVE
C
         IRW=ANSROW+IT2-I
         ICL=NPORT+IT1+IT2-I
         IT3=IRW-1
C
C           J=ROW ZEROING OUT ABOVE PIVOT
C
      DO 264 J=II,IT3
         B=ANS(J,ICL)
C
C           K=COLUMN CHANGING OF JTH ROW
C
      DO 264 K=ICL,ANSCOL
  264 ANS(J,K)=ANS(J,K)-B*ANS(IRW,K)
  266 DO 268 I=II,NBR
      DO 268 J=NPORT1,ANSCOL
  268 IF (ABS(ANS(I,J)).LE.ZERO) ANS(I,J)=0.0
C
C           DPRINT FINAL ANS MATRIX
C
      IF (DEBUG.NE.1) GO TO 270
      CALL DPRNT1 (IT,NPORT1,ANSCOL,II,NBR,HEADER,ANS,ME)
  270 RETURN
C
  272 FORMAT (1H0///)
  274 FORMAT (1H0,10HTREE PORT BRANCHES,/30(1X,I2))
  276 FORMAT (1H0,22HTREE NON-PORT BRANCHES,/30(1X,I2))
  278 FORMAT (1H0,22HLINK NON-PORT BRANCHES,/30(1X,I2))
  280 FORMAT (1H0,18HLINK PORT BRANCHES,/30(1X,I2))
  282 FORMAT (1H0,  5HTP = ,I3/,  6H TN = ,I3/,  6H LN = ,I3/,  6H LP =
     1,I3)
```

```
  284 FORMAT (1H0,  2HBR,40(1X,I2))
  285 FORMAT (///, 18H F3 BEFORE ZEROING)
  288 FORMAT (1X)
  290 FORMAT (1X,10(E11.4,1X))
  292 FORMAT (///, 18H F5 BEFORE ZEROING)
  294 FORMAT (///, 25H ANS MATRIX BEFORE ZEROING)
  296 FORMAT (///, 25H ANS MATRIX AFTER ZEROING)
  298 FORMAT (///, 35H ANS MATRIX WITH D VALUES FILLED IN)
  300 FORMAT (///, 35H ANS MATRIX REDUCED TO ECHELON FORM)
C
      END
      SUBROUTINE DIAECH (NROW,NCOL,A,MU)
C
C*************************************************************************
C      *                                                                 *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:               *
C      *      1. MANIPULATE THE INCIDENCE (A) MATRIX INTO ECHELON FORM   *
C      *                                                                 *
C******  THIS SUB-PROGRAM*S GLOSSARY OF FORTRAN NAMES:                   *
C      *      NROW      : NUMBER OF ROWS IN THE A MATRIX                 *
C      *      NCOL      : NUMBER OF COLUMNS IN THE A MATRIX              *
C      *                                                                 *
C*************************************************************************
C
C         SUBROUTINE DIAECH MANIPULATES MATRIX A INTO ECHELON FORM
C
      INTEGER A,C,G,GPLUS1,P,B
      DIMENSION A(MU,1)
      C=1
      G=1
  102 DO 116 I=G,NROW
         IF (A(I,C).EO.0) GO TO 116
C
C         INTERCHANGE I AND G ROW TO GET NONZERO PIVOT
C
         IF (I.EO.G) GO TO 106
         DO 104 K=C,NCOL
            B=A(I,K)
            A(I,K)=A(G,K)
            A(G,K)=B
  104    CONTINUE
C
C         NORMALIZE ROW TO GET POSITIVE NUMBER FOR PIVOT
C
  106    IF (A(G,C).GT.0) GO TO 110
         DO 108 K=C,NCOL
  108    A(G,K)=-A(G,K)
  110    IF (G.GE.NROW) RETURN
C
C         ZERO COLUMN BELOW PIVOT
C
         GPLUS1=G+1
         DO 114 P=GPLUS1,NROW
            B=A(P,C)
            IF (B.EQ.0) GO TO 114
            DO 112 K=C,NCOL
  112       A(P,K)=-B*A(G,K)+A(P,K)
  114    CONTINUE
         G=G+1
         C=C+1
```

```fortran
      GO TO 102
  115 CONTINUE
      IF (G.GT.NROW) RETURN
      C=C+1
      GO TO 102
C
      END
      SUBROUTINE DPRINT (ANSCOL,ANSROW,ANS,ME)
C
C*********************************************************************
C     *                                                              *
C*****    THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:          *
C     *         1. PRINT THE ENTIRE HYBRID MATRIX FOR DEBUG RUN.     *
C     *                                                              *
C*****    THIS SUB-PROGRAM*S GLOSSARY OF FORTRAN VARIABLES:          *
C     *         ANSCOL    : NUMBER OF COLUMNS IN THE HYBRID(ANS) MATRIX  *
C     *         ANSROW    : NUMBER OF ROWS IN THE HYBRID MATRIX      *
C     *                                                              *
C*********************************************************************
C
C         SUBROUTINE DPRINT DPRINTS THE ENTIRE ANS MATRIX
C
C         PRINTS ANSROW ROWS BY ANSCOL COLUMNS
C
      INTEGER ANSCOL,ANSROW
      DIMENSION ANS(ME,1)
      IT1=1
  102 IT2=ANSCOL
      IF ((IT2-IT1).GT.9) GO TO 106
      IF (IT2.EQ.IT1) RETURN
C
C         LESS THAN 10 COLUMNS LEFT TO PRINT
C
      WRITE (6,110)
      DO 104 I=1,ANSROW
  104 WRITE (6,112) (ANS(I,J),J=IT1,IT2)
      RETURN
  106 IT2=IT1+9
C
C         MORE THAN 10 COLUMNS LEFT TO PRINT
C
      WRITE (6,110)
      DO 108 I=1,ANSROW
  108 WRITE (6,112) (ANS(I,J),J=IT1,IT2)
      IT1=IT2+1
      GO TO 102
C
  110 FORMAT (1X)
  112 FORMAT (1X,10(E11.4,1X))
C
      END
      SUBROUTINE DPRNT1 (NDR,ACL1,ACL2,ARW1,ARW2,HEADER,ANS,ME)
C
C*********************************************************************
C     *                                                              *
C*****    THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:          *
C     *         1. PRINT THE DESIRED PORTION OF THE HYBRID MATRIX.   *
C     *                                                              *
C*****    THIS SUB-PROGRAM*S GLOSSARY OF FORTRAN NAMES:              *
C     *         NDR       : TOTAL NUMBER OF COLUMNS IN THE DESIRED PART  *
```

```
C       *         ACL1         : FIRST COLUMN OF THE DESIRED PART             *
C       *         ACL2         : LAST COLUMN OF THE DESIRED PART              *
C       *         ARW1         : FIRST ROW OF THE DESIRED PART                *
C       *         ARW2         : LAST ROW OF THE DESIRED PART                 *
C       *         HEADER       : COLUMN HEADING VECTOR                        *
C       *         ANS          : HYBRID MATRIX                                *
C       *         ME           : ROW DIMENSION OF ANS IN THE CALLING PROGRAM  *
C       *                                                                     *
C**********************************************************************
C
C          SUBROUTINE DPRNT1 PRINTS ONLY THE DESIRED PART OF THE ANS
C          MATRIX DESCRIBING THE PORT EQUATIONS ALONG WITH THE COLUMN
C          HEADINGS
C
      INTEGER A,HEADER,ACL1,ACL2,ARW1,ARW2,HDR
      DIMENSION HEADER(300)
      DIMENSION ANS(ME,1)
      ITM2=ACL1-1
      IT1=1
  102 IT2=HDR
      IF ((IT2-IT1).GT.19) GO TO 106
      IF (IT2.EQ.IT1) RETURN
C
C          LESS OR EQUAL 10 COLUMNS TO PRINT
C
      WRITE (6,110) (HEADER(I),I=IT1,IT2)
      ITM1=ITM2+1
      DO 104 I=ARW1,ARW2
  104 WRITE (6,112) (ANS(I,J),J=ITM1,ACL2)
      RETURN
  106 IT2=IT1+19
C
C          MORE THAN 10 COLUMNS TO PRINT
C
      WRITE (6,110) (HEADER(I),I=IT1,IT2)
      ITM1=ITM2+1
      ITM2=ITM1+9
      DO 108 I=ARW1,ARW2
  108 WRITE (6,112) (ANS(I,J),J=ITM1,ITM2)
      IT1=IT2+1
      GO TO 102
C
  110 FORMAT (1H0,10(4X,A1,I2,5X))
  112 FORMAT (1H0,10(E11.4,1X))
C
      END
      SUBROUTINE DRAECH (M,N,MARK,ROW1,COL1,AD,MV,ME)
C
C**********************************************************************
C       *                                                             *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:            *
C       *         1. OPERATE ON THE ROWS OF THE HYBRID MATRIX TO REDUCE IT *
C       *            TO ECHELON FORM                                  *
C       *                                                             *
C******  THIS SUB-PROGRAM'S GLOSSARY OF FORTRAN NAMES:                 *
C       *         M            : LAST ROW NUMBER IN ECHELON PART OF HYBRID MAT*
C       *         N            : LAST ROW NUMBER FOR ROW OPERATION      *
C       *         MARK         : LAST COLUMN NUMBER IN ECHELON FORM MATRIX *
C       *         ROW1         : FIRST ROW NUMBER IN ECHELON FORM MATRIX *
C       *         COL1         : FIRST COLUMN NUMBER IN ECHELON FORM MATRIX *
```

```
C      *        ALL OTHER VARIABLE NAMES AS DEFINED IN SUB-PROGRAM AMAIN *
C      *                                                                 *
C*****************************************************************************
C
      DIMENSION AD(ME,1)
      INTEGER C,G,GPLUS1,P,ROW1,COL1
C
C          DRAECH PERFORMS ROW OPERATIONS ON A TO REDUCE A TO ECHELON FOR
C
C          COLUMNS COL1 TO MARK ARE REDUCED TO ROW ECHELON FORM WHILE THE
C          ROW OPERATIONS ARE CARRIED OUT ON THE ROWS FROM MARK + 1 TO N.
C          ROWS ROW1 TO N ARE REDUCED TO ROW ECHELON FORM
C          G IS THE ROW IN WHICH WE ARE DETERMINING THE PIVOT POINT
C          C IS THE COLUMN IN WHICH WE ARE DETERMINING THE PIVOT POINT
C
      C=COL1-1
      G=ROW1
  102 IF (C.EQ.MARK) RETURN
      C=C+1
C
C          FIND THE MAX NONZERO ELEMENT IN THE C COLUMN BELOW AND
C          INCLUDING PIVOT
C
      I=0
      ZERO=1.000E-15
      TMZ=0.0
      DO 104 J=G,N
         IF (ABS(AD(J,C)).LE.ZERO) AD(J,C)=0.0
         IF (ABS(AD(J,C)).LE.TMZ) GO TO 104
         TMZ=ABS(AD(J,C))
         I=J
  104 CONTINUE
      IF (TMZ.EQ.0.0) GO TO 102
C
C          IF THE NONZERO ELEMENT IS IN THE PIVOT ROW, DO NOT EXCHANGE
C          ROWS
C
      IF (I.EQ.G) GO TO 103
C
C          EXCHANGE PIVOT ROW WITH ROW HAVING NONZERO ELEMENT IN PIVOT
C          COLUMN
C
      DO 106 K=C,N
         D=AD(I,K)
         AD(I,K)=AD(G,K)
  106 AD(G,K)=D
C
C          CHECK IF PIVOT POINT ALREADY NORMALIZED TO 1
C
  103 IF (AD(G,C).EQ.1.) GO TO 112
C
C          NORMALIZE PIVOT ROW
C
      ALPHA=AD(G,C)
      DO 110 K=C,N
         AD(G,K)=AD(G,K)/ALPHA
  110 IF (ABS(AD(G,K)).LE.ZERO) AD(G,K)=0.0
C
C          CHECK IF JUST NORMALIZED PIVOT IN LAST ROW
C
```

```
  112 IF (G.GE.M) RETURN
C
C          ZERO THE ELEMENTS BELOW THE PIVOT
C
      GPLUS1=G+1
      DO 116 P=GPLUS1,M
         B=AD(P,C)
         IF (ABS(AD(P,C)).LE.ZERO) AD(P,C)=0.0
         IF (ABS(AD(P,C)).EQ.0.0) GO TO 116
         DO 114 K=C,N
  114    AD(P,K)=-B*AD(G,K)+AD(P,K)
  116 CONTINUE
      IF (G.GE.M) RETURN
      G=G+1
      GO TO 102
C
      END
      SUBROUTINE ESTATE (NPORT1,ANSCOL,II,NBR,NSV,DEBUG,A,B,C,D,VALUE,AN
     1S,ME,MS,MP)
C
C*********************************************************************
C     *                                                              *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTIONS:          *
C     *        1. OBTAIN THE MATRICES IN THE STATE SPACE REPRESENTATION *
C     *           FOR THE AUGMENTED LINEAR NETWORK.                  *
C     *        2. PRINT THE STATE SPACE DESCRIPTION, IF REQUESTED.   *
C     *                                                              *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:               *
C     *        NPORT1    : ADDRESS FOR LOCATING FIRST COLUMN OF MATRIX A*
C     *        ANSCOL    : ADDRESS FOR LOCATING FIRST COLUMN OF MATRIX B*
C     *        II        : ADDRESS FOR LOCATING FIRST ROW OF MATRIX A  *
C     *        NBR       : TOTAL NUMBER OF BRANCHES IN LINEAR CIRCUIT  *
C     *        NSV       : TOTAL NUMBER OF STATE VARIABLES             *
C     *        DEBUG     : FLAG VARIABLE FOR PRINTING STATE EQUATIONS  *
C     *        A         : MATRIX A IN STATE SPACE DESCRIPTION         *
C     *        B         : MATRIX B IN STATE SPACE DESCRIPTION         *
C     *        C         : MATRIX C IN STATE SPACE DESCRIPTION         *
C     *        D         : MATRIX D IN STATE SPACE DESCRIPTION         *
C     *        VALUE     : ARRAY OF ELEMENT VALUES                     *
C     *        ANS       : HYBRID MATRIX                               *
C     *                                                              *
C*********************************************************************
C
      INTEGER ANSCOL,CONN,DEBUG
      DIMENSION VALUE(1)
      DIMENSION ANS(ME,1)
      DIMENSION A(MS,1), B(MS,1), C(MP,1), D(MP,1)
      DIMENSION DENOM(20)
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NDCS,NCS
      NCP1=NCAP+1
      IF (NCAP.EQ.0) GO TO 104
      DO 102 I=1,NCAP
  102 DENOM(I)=VALUE(I)
  104 IF (NIND.EQ.0) GO TO 108
      K=NCAP+NDVS+NRES
      DO 106 I=NCP1,NSV
         K=K+1
  106 DENOM(I)=VALUE(K)
  108 NEQNS=NBR+1-II
C
```

AD-A088 422 PURDUE UNIV LAFAYETTE IND F/G 9/3
PRANC: PROGRAM FOR ANALYZING NONLINEAR CIRCUITS.(U)
MAY 80 H K THAPAR, B J LEON F30602-78-C-0102
UNCLASSIFIED RADC-TR-80-139 NL

3 OF 3
AD A 088422

END
DATE FILMED
9-80
DTIC

```
C          FILL MATRIX A
C
      N1=II
      N2=N1+NCAP+NIND-1
      N3=NPORT1+NEQNS
      N4=N3+NCAP+NIND-1
      IF (N2.LT.N1) GO TO 123
      I1=0
      DO 110 I=N1,N2
         I1=I1+1
         J1=0
      DO 110 J=N3,N4
         J1=J1+1
  110 A(I1,J1)=-ANS(I,J)/DENOM(I1)
C
C          FILL MATRIX B
C
      N5=ANSCOL-NCS+1
      N6=ANSCOL
      I1=0
      DO 114 I=N1,N2
         I1=I1+1
         J1=0
         DO 112 J=N5,N6
            J1=J1+1
  112    B(I1,J1)=-ANS(I,J)/DENOM(I1)
  114 CONTINUE
C
C*****FILL MATRIX C
C
      I1=0
      N1=N2+1
      N2=N2+NCS
      N3=ANSCOL-NCS-NSV+1
      N4=N3+NSV
      DO 116 I=N1,N2
         I1=I1+1
         J1=0
      DO 116 J=N3,N4
         J1=J1+1
         C(I1,J1)=-ANS(I,J)
  116 CONTINUE
C
C*****FILL MATRIX D
C
      N5=ANSCOL-NCS+1
      N6=N5+NCS
      I1=0
      DO 118 I=N1,N2
         I1=I1+1
         J1=0
      DO 118 J=N5,N6
         J1=J1+1
         D(I1,J1)=-ANS(I,J)
  118 CONTINUE
C
C          PRINT MATRICES A, B,C , D
C
      IF (DEBUG.NE.1) GO TO 123
      WRITE (6,130)
```

```
      DO 120 I=1,NSV
  120 WRITE (6,132) (A(I,J),J=1,NSV)
      WRITE (6,134)
      DO 122 I=1,NSV
  122 WRITE (6,132) (B(I,J),J=1,NCS)
      WRITE (6,136)
      DO 124 I=1,NCS
  124 WRITE (6,132) (C(I,J),J=1,NSV)
      WRITE (6,138)
      DO 126 I=1,NCS
  126 WRITE (6,132) (D(I,J),J=1,NCS)
  128 RETURN
C
  130 FORMAT (1H1,9H MATRIX A)
  132 FORMAT (X,11(E10.3,2X))
  134 FORMAT (/,9H MATRIX B)
  136 FORMAT (/,9H MATRIX C)
  138 FORMAT (/,9H MATRIX D)
C
      END
```

```
      SUBROUTINE FBALNC (A,N,IA,B,K,L)
C
C*********************************************************************
C     *                                                              *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:           *
C     *        1. BALANCE A REAL MATRIX A.                           *
C     *                                                              *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:               *
C     *        A          : MATRIX TO BE BALANCED                    *
C     *        N          : DIMENSION OF MATRIX A                    *
C     *        IA         : ROW DIMENSION OF A                       *
C     *        D          : ARRAY CONTAINING INFORMATION ABOUT PERMUTATIO*
C     *                     AND SCALE FACTORS                        *
C     *        K,L        : INTEGERS SUCH THAT A(I,J)=0 IF (1) I GT J AND*
C     *                     (2) J=1,2,...,K-1 OR I=L+1,...,N         *
C     *                                                              *
C*********************************************************************
C
      DIMENSION A(IA,1), D(1)
      DATA B/16.0/,B2/256.0/
      DATA ZERO/0.0/,ONE/1.0/,P95/.95/
C
C******                             REDUCE NORM A BY DIAGONAL SIMILARITY
C******                             TRANSFORMATION STORED IN D
C
      L1=1
      K1=N
C
C******                             SEARCH FOR ROWS ISOLATING AN EIGEN-
C******                             VALUE AND PUSH THEM DOWN
C
  101 K1P1=K1+1
      IF (K1.LT.1) GO TO 107
      K11=K1
      DO 106 JJ=1,K11
         J=K1P1-JJ
         R=ZERO
         DO 102 I=1,K1
            IF (I.EQ.J) GO TO 102
            R=R+ABS(A(J,I))
  102    CONTINUE
         IF (R.NE.ZERO) GO TO 106
         D(K1)=J
         IF (J.EQ.K1) GO TO 105
         DO 103 I=1,K1
            F=A(I,J)
            A(I,J)=A(I,K1)
            A(I,K1)=F
  103    CONTINUE
         DO 104 I=L1,N
            F=A(J,I)
            A(J,I)=A(K1,I)
            A(K1,I)=F
  104    CONTINUE
  105    K1=K1-1
         GO TO 101
  106 CONTINUE
C
C******                             SEARCH FOR COLUMNS ISOLATING AN
C******                             EIGENVALUE AND PUSH THEM LEFT
```

```
C
  107 IF (K1.LT.L1) GO TO 113
      LL=L1
      DO 112 J=LL,K1
         C=ZERO
         DO 108 I=L1,K1
            IF (I.EQ.J) GO TO 108
            C=C+ABS(A(I,J))
  108    CONTINUE
         IF (C.NE.ZERO) GO TO 112
         D(L1)=J
         IF (J.EQ.L1) GO TO 111
         DO 109 I=1,K1
            F=A(I,J)
            A(I,J)=A(I,L1)
            A(I,L1)=F
  109    CONTINUE
         DO 110 I=L1,N
            F=A(J,I)
            A(J,I)=A(L1,I)
            A(L1,I)=F
  110    CONTINUE
  111    L1=L1+1
         GO TO 107
  112 CONTINUE
C
C******                                  NOW BALANCE THE SUBMATRIX IN ROWS
C******                                  L1 THROUGH K1
C
  113 K=L1
      L=K1
      IF (K1.LT.L1) GO TO 115
      DO 114 I=L1,K1
         D(I)=ONE
  114 CONTINUE
  115 NOCONV=0
      IF (K1.LT.L1) GO TO 124
      DO 123 I=L1,K1
         C=ZERO
         R=ZERO
         DO 116 J=L1,K1
            IF (J.EQ.I) GO TO 116
            C=C+ABS(A(J,I))
            R=R+ABS(A(I,J))
  116    CONTINUE
         G=R/B
         F=ONE
         S=C+R
  117    IF (C.GE.G) GO TO 118
         F=F*B
         C=C*B2
         GO TO 117
  118    G=R*B
  119    IF (C.LT.G) GO TO 120
         F=F/B
         C=C/B2
         GO TO 119
C
C******                                  NOW BALANCE
C
```

```
  120     IF ((C+R)/F.GE.P95*S) GO TO 123
          G=ONE/F
          D(I)=D(I)*F
          NOCONV=1
          DO 121 J=L1,N
             A(I,J)=A(I,J)*G
  121     CONTINUE
          DO 122 J=1,K1
             A(J,I)=A(J,I)*F
  122     CONTINUE
  123 CONTINUE
  124 IF (NOCONV.EQ.1) GO TO 115
      RETURN
C
      END
      SUBROUTINE FEVEV (A,N,IA,W,Z,WK,IER)
C
C***********************************************************************
C     *                                                                *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:             *
C     *        1. ACT AS THE EXECUTIVE CALLING PROGRAM FOR OBTAINING   *
C     *           THE EIGENVALUES-EIGENVECTORS OF A REAL MATRIX.       *
C     *                                                                *
C******  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINES:              *
C     *        1. FBALNC                                               *
C     *        2. FRDHSS                                               *
C     *        3. FBKXM1                                               *
C     *        4. FBKXM2                                               *
C     *        5. FQRALG                                               *
C     *        6. FERTST                                               *
C     *                                                                *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                 *
C     *        A          : MATRIX WHOSE EIGENVALUES-EIGENVECTORS ARE TO *
C     *                     TO BE FOUND                                *
C     *        N          : DIMENSION OF MATRIX A                      *
C     *        IA         : ROW DIMENSION OF A                         *
C     *        W          : ARRAY CONTAINING THE EIGENALUES            *
C     *        Z          : MODAL MATRIX                               *
C     *        WK         : WORK ARRAY                                 *
C     *        IER        : ERROR PARAMETER                            *
C     *                                                                *
C***********************************************************************
C
      DIMENSION A(IA,1), W(1), WK(N,1), Z(1)
      DATA ZERO,ONE/0.0,1.0/
C
C******                           INITIALIZE ERROR PARAMETERS
C
      IER=0
      JER=0
      IZ=IA
      IZ2=IZ+IZ
C
C******                           PACK A INTO AN N BY N ARRAY
C
      K=1
      L=1
      DO 105 J=1,N
      DO 105 I=1,N
         A(K,L)=A(I,J)
```

```
         K=K+1
         IF (K.GT.IA) K=1
         IF (K.EQ.1) L=L+1
  105 CONTINUE
      N1=1
      N2=N1+1
C
C******                        BALANCE THE INPUT A
C
      CALL FBALNC (A,N,N,WK(1,N1),K,L)
C
C******                        IF L = 0, A IS ALREADY IN HESSENBERG
C******                          FORM
C
      CALL FRDHSS (A,K,L,N,N,WK(1,N2))
C
C******                        SET Z IDENTITY MATRIX
C
      II=1
      JJ=1
      NP1=N+1
      DO 115 I=1,N
         DO 110 J=1,N
            Z(II)=ZERO
            II=II+1
  110    CONTINUE
         Z(JJ)=ONE
         JJ=JJ+NP1
  115 CONTINUE
      CALL FBKXM1 (Z,A,WK(1,N2),N,N,K,L)
      IIZ=N
      CALL FQRALG (A,N,N,K,L,W(1),W(N+1),Z,IIZ,JER)
      IF (JER.GT.128) GO TO 120
      CALL FBKXM2 (WK(1,N1),Z,K,L,N,N,N)
C
C******                        CONVERT W (EIGENVALUES) TO COMPLEX
C******                          FORMAT
C
  120 DO 125 I=1,N
         NPI=N+I
         WK(I,N1)=W(NPI)
  125 CONTINUE
      JW=N+N
      J=N
      DO 130 I=1,N
         W(JW-1)=W(J)
         W(JW)=WK(J,N1)
         JW=JW-2
         J=J-1
  130 CONTINUE
C
C******                        CONVERT Z (EIGENVECTORS) TO COMPLEX
C******                          FORMAT Z(IZ,N)
C
      J=N
  135 IF (J.LT.1) GO TO 160
      IF (W(J+J).EQ.ZERO) GO TO 150
C
C******                        MOVE PAIR OF COMPLEX CONJUGATE
C******                          EIGENVECTORS
```

```
C
      IS=IZ2*(J-1)+1
      IG=N*(J-2)+1
      IGZ=IG+N
C
C******                           MOVE COMPLEX CONJUGATE EIGENVECTOR
C
      DO 140 I=1,N
         Z(IS)=Z(IG)
         Z(IS+1)=-Z(IGZ)
         IS=IS+2
         IG=IG+1
         IGZ=IGZ+1
  140 CONTINUE
C
C******                           MOVE COMPLEX EIGENVECTOR
C
      IS=IZ2*(J-2)+1
      IG=IS+IZ2
      DO 145 I=1,N
         Z(IS)=Z(IG)
         Z(IS+1)=-Z(IG+1)
         IS=IS+2
         IG=IG+2
  145 CONTINUE
      J=J-2
      GO TO 135
C
C******                           MOVE REAL EIGENVECTOR
C
  150 IS=IZ2*(J-1)+N+N
      IG=N*J
      DO 155 I=1,N
         Z(IS-1)=Z(IG)
         Z(IS)=ZERO
         IS=IS-2
         IG=IG-1
  155 CONTINUE
      J=J-1
      GO TO 135
C
C******WRITE ERROR MESSAGES, IF ANY
C
  160 IF (IER.NE.0) CALL FERTST (IER,6HFEVEV )
      IF (JER.EQ.0) GO TO 165
      IER=JER
      CALL FERTST (IER,6HFEVEV )
  165 RETURN
C
      END
      SUBROUTINE FBKXM1 (Z,H,D,MM,IZH,K,L)
C
C*********************************************************************
C     *                                                              *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:           *
C     *       1. BACKTRANSFORM THE EIGENVECTORS OF THE UPPER HESSENBERG*
C     *          MATRIX.                                             *
C     *                                                              *
C******  THIS SUB-PROGRAM≠S GLOSSARY OF FORTRAN NAMES:               *
C     *       Z          : EIGENVECTORS OF MATRIX A                  *
```

```
C     *        H          : SUB-DIAGONAL ELEMENTS USED FOR STORING BACK-
C     *                     TRANSFORMATION INFORMATION
C     *        D          : DETAILS OF THE TRANSFORMATION
C     *        MM         : NUMBER OF COLUMNS IN MATRIX Z
C     *        IZH        : ROW DIMENSION OF MATRICES Z AND H
C     *        K,L        : SAME AS IN SUBROUTINE FBKXM1
C     *
C***********************************************************************
C
      DIMENSION Z(IZH,1), H(IZH,1), D(1)
      DATA ZERO,ONE/0.0,1.0/
      LM2=L-2
      IF (LM2.LT.K) GO TO 107
      LTEMP=LM2+K
      DO 106 KI=K,LM2
         M=LTEMP-KI
         MA=M+1
         T=H(MA,M)
         IF (T.EQ.ZERO) GO TO 106
         T=T*D(MA)
         MP2=M+2
         IF (MP2.GT.L) GO TO 102
         DO 101 I=MP2,L
            D(I)=H(I,M)
  101    CONTINUE
  102    IF (MA.GT.L) GO TO 106
         TINV=ONE/T
         DO 105 J=1,MM
            G=ZERO
            DO 103 I=MA,L
               G=G+D(I)*Z(I,J)
  103       CONTINUE
            G=G*TINV
            DO 104 I=MA,L
               Z(I,J)=Z(I,J)+G*D(I)
  104       CONTINUE
  105    CONTINUE
  106 CONTINUE
  107 RETURN
C
      END
      SUBROUTINE FBKXM2 (D,Z,K,L,MM,N,IZ)
C
C***********************************************************************
C     *
C******    THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:
C     *        1. BACKTRANSFORM THE EIGENVECTORS OF A BALANCED MATRIX
C     *
C******    THIS SUB-PROGRAM≠S GLOSSARY OF FORTRAN NAMES:
C     *        D          : INFORMATION ON THE DETAILS OF TRANSFORMATION
C     *        Z          : AT ENTRANCE: MODAL MATRIX TO BE TRANSFORMED
C     *                     AT EXIT, TRANSFORMED MODAL MATRIX
C     *        K          : ROW,COLUMN INDEX OF STARTING ELEMENT TO BE
C     *                     TRANSFORMED
C     *        L          : ROW,COLUMN INDEX OF LAST ELEMENT TO BE TRANS-
C     *                     FORMED
C     *        MM         : NUMBER OF COLUMNS IN MATRIX Z
C     *        N          : NUMBER OF ROWS IN Z = LENGTH OF VECTOR D
C     *        IZ         : ROW DIMENSION OF Z
C     *
```

```
C***********************************************************************
C
      DIMENSION Z(IZ,1), D(1)
C
C******                            COLUMN SCALE Z BY APPROPRIATE
C******                              D VALUE
C
      DO 101 I=K,L
         S=D(I)
      DO 101 J=1,MM
         Z(I,J)=Z(I,J)*S
  101 CONTINUE
C
C******                            INTERCHANGE ROWS IF PERMUTATIONS
C******                              OCCURRED IN FBALNC
C
      IF (K.EQ.1) GO TO 104
      KM1=K-1
      DO 103 I=1,KM1
         II=K-I
         JJ=D(II)
         IF (II.EQ.JJ) GO TO 103
         DO 102 J=1,MM
            S=Z(II,J)
            Z(II,J)=Z(JJ,J)
            Z(JJ,J)=S
  102    CONTINUE
  103 CONTINUE
  104 IF (L.EQ.N) GO TO 107
      LP1=L+1
      DO 106 II=LP1,N
         JJ=D(II)
         IF (II.EQ.JJ) GO TO 106
         DO 105 J=1,MM
            S=Z(II,J)
            Z(II,J)=Z(JJ,J)
            Z(JJ,J)=S
  105    CONTINUE
  106 CONTINUE
  107 RETURN
C
      END
      SUBROUTINE FERTST (IER,NAME)
C
C***********************************************************************
C     *  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:             *
C     *       1. PRINT ERROR MESSAGE ARISING IN FEVEV OR FQRALG ROUTINE*
C     *                                                                *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                 *
C     *       IER       : ERROR PARAMETER VALUE                        *
C     *       NAME      : NAME OF THE CALLING SUB-PROGRAM              *
C     *                                                                *
C***********************************************************************
C
      DIMENSION ITYP(2,4), IBIT(4)
      INTEGER WARN,WARF,TERM,PRINTR
      EQUIVALENCE (IBIT(1),WARN), (IBIT(2),WARF), (IBIT(3),TERM)
      DATA ITYP/10HWARNING   ,10H          ,10HWARNING(WI,10HTH FIX)   ,
     110HTERMINAL  ,10H          ,10HNON-DEFINE,10HD         /,IBIT/32,6
     24,128,0/
```

```
      IERR=IER
      IF (IERR.GE.WARN) GO TO 101
C
C******                        NON-DEFINED
C
      IERK=4
      GO TO 104
  101 IF (IERR.LT.TERM) GO TO 102
C
C******                        TERMINAL
C
      IERK=3
      GO TO 104
  102 IF (IERR.LT.WARF) GO TO 103
C
C******                        WARNING(WITH FIX)
C
      IERK=2
      GO TO 104
C
C******                        WARNING
C
  103 IERK=1
C
C******                        EXTRACT *N*
C
  104 IERR=IERR-IBIT(IERK)
C
C******                        PRINT ERROR MESSAGE
C
      WRITE (6,105) (ITYP(I,IERK),I=1,2),NAME,IERR,IER
      RETURN
C
  105 FORMAT (1H0,2A10,4X,A6,4X,I2,8H (IER = ,I3,1H))
C
      END
      SUBROUTINE FORALG (HS,N,IH,K,L,WRL,WIM,Z,IZ,IER)
C
C***********************************************************************
C     *                                                                *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:             *
C     *       1. FIND THE EIGENVALUES AND EIGENVECTORS OF THE UPPER    *
C     *          HESSENBERG MATRIX.                                    *
C     *                                                                *
C******  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINE:               *
C     *       1. FERTST                                                *
C     *                                                                *
C******  THIS SUB-PROGRAM≠S GLOSSARY OF FORTRAN NAMES:                 *
C     *  ALL VARIABLE NAMES AND ARRAYS ARE AS DEFINED IN SUBROUTINES   *
C     *  FEVEV,FBALNC,FBKXM1,FBKXM2.                                   *
C     *                                                                *
C***********************************************************************
C
      DIMENSION HS(IH,N), WRL(N), WIM(N), Z(IZ,N), T3(2)
      LOGICAL NTLS
      COMPLEX Z3
      EQUIVALENCE (Z3,T3(1))
      DATA RDELP/16414000000000000000B/
      DATA P4/0.4375/,P5/0.5/,P7/0.75/,ZERO/0.0/,ONE/1.0/
      IER=0
```

```
C
C******                           STORE ROOTS ISOLATED BY FBALNC
C
      DO 101 I=1,N
         IF (I.GE.K.AND.I.LE.L) GO TO 101
         WRL(I)=HS(I,I)
         WIM(I)=ZERO
  101 CONTINUE
      IEN=L
      T=ZERO
C
C******                           SEARCH FOR NEXT EIGENVALUES
C
  102 IF (IEN.LT.K) GO TO 123
      ITS=0
      NA=IEN-1
      IENM2=NA-1
C
C******                           LOOK FOR SINGLE SMALL SUB-DIAGONAL
C******                           ELEMENT
C
  103 NPL=IEN+K
      DO 104 LL=K,IEN
         LB=NPL-LL
         IF (LB.EQ.K) GO TO 105
         IF (ABS(HS(LB,LB-1)).LE.RDELP*(ABS(HS(LB-1,LB-1))+ABS(HS(LB,LB)
     1   ))) GO TO 105
  104 CONTINUE
C
C******
C
  105 X=HS(IEN,IEN)
      IF (LB.EQ.IEN) GO TO 121
      Y=HS(NA,NA)
      W=HS(IEN,NA)*HS(NA,IEN)
      IF (LB.EQ.NA) GO TO 122
      IF (ITS.EQ.30) GO TO 151
C
C******                           FORM SHIFT
C
      IF (ITS.NE.10.AND.ITS.NE.20) GO TO 107
      T=T+X
      DO 106 I=K,IEN
         HS(I,I)=HS(I,I)-X
  106 CONTINUE
      S=ABS(HS(IEN,NA))+ABS(HS(NA,IENM2))
      X=P7*S
      Y=X
      W=-P4*S*S
  107 ITS=ITS+1
C
C******                           LOOK FOR TWO CONSECUTIVE SMALL
C******                           SUB-DIAGONAL ELEMENTS
C
      NAML=IENM2+LB
      DO 108 MM=LB,IENM2
         M=NAML-MM
         ZZ=HS(M,M)
         R=X-ZZ
         S=Y-ZZ
```

```
      P=(R*S-W)/HS(M+1,M)+HS(M,M+1)
      Q=HS(M+1,M+1)-ZZ-R-S
      R=HS(M+2,M+1)
      S=ABS(P)+ABS(Q)+ABS(R)
      P=P/S
      Q=Q/S
      R=R/S
      IF (M.EQ.LB) GO TO 109
      IF (ABS(HS(M,M-1))*(ABS(Q)+ABS(R)).LE.RDELP*ABS(P)*(ABS(HS(M-1,
     1 M-1))+ABS(ZZ)+ABS(HS(M+1,M+1)))) GO TO 109
  108 CONTINUE
  109 MP2=M+2
      DO 110 I=MP2,IEN
         HS(I,I-2)=ZERO
         IF (I.EQ.MP2) GO TO 110
         HS(I,I-3)=ZERO
  110 CONTINUE
C
C******                        DOUBLE QR STEP INVOLVING ROWS
C******                        L TO EN AND COLUMNS M TO EN
C
      DO 120 KA=M,NA
         NTLS=KA.NE.NA
         IF (KA.EQ.M) GO TO 111
         P=HS(KA,KA-1)
         Q=HS(KA+1,KA-1)
         R=ZERO
         IF (NTLS) R=HS(KA+2,KA-1)
         X=ABS(P)+ABS(Q)+ABS(R)
         IF (X.EQ.ZERO) GO TO 120
         P=P/X
         Q=Q/X
         R=R/X
  111    CONTINUE
         S=SIGN(SQRT(P*P+Q*Q+R*R),P)
         IF (KA.EQ.M) GO TO 112
         HS(KA,KA-1)=-S*X
         GO TO 113
  112    IF (LB.NE.M) HS(KA,KA-1)=-HS(KA,KA-1)
  113    P=P+S
         X=P/S
         Y=Q/S
         ZZ=R/S
         Q=Q/P
         R=R/P
C
C******                        ROW MODIFICATION
C
         DO 115 J=KA,N
            P=HS(KA,J)+Q*HS(KA+1,J)
            IF (.NOT.NTLS) GO TO 114
            P=P+R*HS(KA+2,J)
            HS(KA+2,J)=HS(KA+2,J)-P*ZZ
  114       HS(KA+1,J)=HS(KA+1,J)-P*Y
            HS(KA,J)=HS(KA,J)-P*X
  115    CONTINUE
         J=MIN0(IEN,KA+3)
C
C******                        COLUMN MODIFICATION
C
```

```
          DO 117 I=1,J
             P=X*HS(I,KA)+Y*HS(I,KA+1)
             IF (.NOT.NTLS) GO TO 116
             P=P+ZZ*HS(I,KA+2)
             HS(I,KA+2)=HS(I,KA+2)-P*R
  116        HS(I,KA+1)=HS(I,KA+1)-P*Q
             HS(I,KA)=HS(I,KA)-P
  117     CONTINUE
          IF (IZ.LT.N) GO TO 120
C
C******                                  ACCUMULATE TRANSFORMATIONS
C
          DO 119 I=K,L
             P=X*Z(I,KA)+Y*Z(I,KA+1)
             IF (.NOT.NTLS) GO TO 118
             P=P+ZZ*Z(I,KA+2)
             Z(I,KA+2)=Z(I,KA+2)-P*R
  118        Z(I,KA+1)=Z(I,KA+1)-P*Q
             Z(I,KA)=Z(I,KA)-P
  119     CONTINUE
  120 CONTINUE
      GO TO 103
C
C******                                  ONE ROOT FOUND
C
  121 HS(IEN,IEN)=X+T
      WRL(IEN)=HS(IEN,IEN)
      WIM(IEN)=ZERO
      IEN=NA
      GO TO 102
C
C******                                  TWO ROOTS FOUND
C
  122 P=(Y-X)*P5
      Q=P*P+W
      ZZ=SQRT(ABS(Q))
      HS(IEN,IEN)=X+T
      X=HS(IEN,IEN)
      HS(NA,NA)=Y+T
      IF (Q.LT.ZERO) GO TO 126
C
C******                                  REAL PAIR
C
      ZZ=P+SIGN(ZZ,P)
      WRL(NA)=X+ZZ
      WRL(IEN)=WRL(NA)
      IF (ZZ.NE.ZERO) WRL(IEN)=X-W/ZZ
      WIM(NA)=ZERO
      WIM(IEN)=ZERO
      X=HS(IEN,NA)
      R=SQRT(X*X+ZZ*ZZ)
      P=X/R
      Q=ZZ/R
C
C******                                  ROW MODIFICATION
C
      DO 123 J=NA,N
         ZZ=HS(NA,J)
         HS(NA,J)=Q*ZZ+P*HS(IEN,J)
         HS(IEN,J)=Q*HS(IEN,J)-P*ZZ
```

```
  123 CONTINUE
C
C******                                     COLUMN MODIFICATION
C
      DO 124 I=1,IEN
         ZZ=HS(I,NA)
         HS(I,NA)=Q*ZZ+P*HS(I,IEN)
         HS(I,IEN)=Q*HS(I,IEN)-P*ZZ
  124 CONTINUE
      IF (IZ.LT.N) GO TO 127
C
C******                                     ACCUMULATE TRANSFORMATIONS
C
      DO 125 I=K,L
         ZZ=Z(I,NA)
         Z(I,NA)=Q*ZZ+P*Z(I,IEN)
         Z(I,IEN)=Q*Z(I,IEN)-P*ZZ
  125 CONTINUE
      GO TO 127
C
C******                                     COMPLEX PAIR
C
  126 WRL(NA)=X+P
      WRL(IEN)=X+P
      WIM(NA)=ZZ
      WIM(IEN)=-ZZ
  127 IEN=IENM2
      GO TO 102
C
C******                                     ALL ROOTS FOUND, NOW
C******                                     BACKSUBSTITUTE
C
  128 IF (IZ.LT.N) GO TO 156
      RNORM=ZERO
      KA=1
      DO 130 I=1,N
         DO 129 J=KA,N
            RNORM=RNORM+ABS(HS(I,J))
  129    CONTINUE
         KA=I
  130 CONTINUE
      IF (RNORM.EQ.ZERO) GO TO 156
      DO 145 NN=1,N
         IEN=N+1-NN
         P=WRL(IEN)
         Q=WIM(IEN)
         NA=IEN-1
         IF (Q.GT.ZERO) GO TO 145
         IF (Q.LT.ZERO) GO TO 137
C
C******                                     REAL VECTOR
C
         M=IEN
         HS(IEN,IEN)=ONE
         IF (NA.EQ.0) GO TO 145
         DO 136 II=1,NA
            I=IEN-II
            W=HS(I,I)-P
            R=HS(I,IEN)
            IF (M.GT.NA) GO TO 132
```

```
            DO 131 J=M,NA
               R=R+HS(I,J)*HS(J,IEN)
  131       CONTINUE
  132       IF (WIM(I).GE.ZERO) GO TO 133
            ZZ=W
            S=R
            GO TO 136
  133       M=I
            IF (WIM(I).NE.ZERO) GO TO 134
            T=W
            IF (W.EQ.ZERO) T=RDELP*RNORM
            HS(I,IEN)=-R/T
            GO TO 136
C
C******                               SOLVE REAL EQUATIONS
C
  134       X=HS(I,I+1)
            Y=HS(I+1,I)
            Q=(WRL(I)-P)*(WRL(I)-P)+WIM(I)*WIM(I)
            T=(X*S-ZZ*R)/Q
            HS(I,IEN)=T
            IF (ABS(X).LE.ABS(ZZ)) GO TO 135
            HS(I+1,IEN)=(-R-W*T)/X
            GO TO 136
  135       HS(I+1,IEN)=(-S-Y*T)/ZZ
  136    CONTINUE
C
C******                               END REAL VECTOR
C
         GO TO 145
C
C******                               LAST VECTOR COMPONENT CHOSEN
C******                                 IMAGINARY SO THAT EIGENVECTOR
C******                                 MATRIX IS TRIANGULAR
C
  137    M=NA
C
C******                               COMPLEX VECTOR
C
         IF (ABS(HS(IEN,NA)).LE.ABS(HS(NA,IEN))) GO TO 138
         HS(NA,NA)=Q/HS(IEN,NA)
         HS(NA,IEN)=-(HS(IEN,IEN)-P)/HS(IEN,NA)
         GO TO 139
  138    CONTINUE
         T3=CMPLX(ZERO,-HS(NA,IEN))/CMPLX(HS(NA,NA)-P,Q)
         HS(NA,NA)=T3(1)
         HS(NA,IEN)=T3(2)
  139    HS(IEN,NA)=ZERO
         HS(IEN,IEN)=ONE
         IENM2=NA-1
         IF (IENM2.EQ.0) GO TO 145
         DO 144 II=1,IENM2
            I=NA-II
            W=HS(I,I)-P
            RA=ZERO
            SA=HS(I,IEN)
            DO 140 J=M,NA
               RA=RA+HS(I,J)*HS(J,NA)
               SA=SA+HS(I,J)*HS(J,IEN)
  140       CONTINUE
```

```
      IF (NIM(I).GE.ZERO) GO TO 141
      ZZ=W
      R=RA
      S=SA
      GO TO 144
141   M=I
      IF (NIM(I).NE.ZERO) GO TO 142
      Z3=CMPLX(-RA,-SA)/CMPLX(W,Q)
      HS(I,NA)=T3(1)
      HS(I,IEN)=T3(2)
      GO TO 144
C
C******                       SOLVE COMPLEX EQUATIONS
C
142   X=HS(I,I+1)
      Y=HS(I+1,I)
      VR=(WRL(I)-P)*(WRL(I)-P)+NIM(I)*NIM(I)-Q*Q
      VI=(WRL(I)-P)*Q
      VI=VI+VI
      IF (VR.EQ.ZERO.AND.VI.EQ.ZERO) VR=RDELP*RNORM*(ABS(W)+ABS(Q)
     1+ABS(X)+ABS(Y)+ABS(ZZ))
      Z3=CMPLX(X*R-ZZ*RA+Q*SA,X*S-ZZ*SA-Q*RA)/CMPLX(VR,VI)
      HS(I,NA)=T3(1)
      HS(I,IEN)=T3(2)
      IF (ABS(X).LE.ABS(ZZ)+ABS(Q)) GO TO 143
      HS(I+1,NA)=(-RA-W*HS(I,NA)+Q*HS(I,IEN))/X
      HS(I+1,IEN)=(-SA-W*HS(I,IEN)-Q*HS(I,NA))/X
      GO TO 144
143   CONTINUE
      Z3=CMPLX(-R-Y*HS(I,NA),-S-Y*HS(I,IEN))/CMPLX(ZZ,Q)
      HS(I+1,NA)=T3(1)
      HS(I+1,IEN)=T3(2)
144   CONTINUE
C
C******                       END COMPLEX VECTOR
C
  145 CONTINUE
C
C******                       END BACKSUBSTITUTION
C******                       VECTORS OF ISOLATED ROOTS
C
      DO 147 I=1,N
         IF (I.GE.K.AND.I.LE.L) GO TO 147
         DO 146 J=I,N
            Z(I,J)=HS(I,J)
  146    CONTINUE
  147 CONTINUE
      IF (L.EQ.0) GO TO 155
C
C******                       MULTIPLY BY TRANSFORMATION MATRIX
C
      DO 150 JJ=K,N
         J=N+K-JJ
         M=MIN0(J,L)
         DO 149 I=K,L
            ZZ=ZERO
            DO 148 KA=K,M
               ZZ=ZZ+Z(I,KA)*HS(KA,J)
148         CONTINUE
            Z(I,J)=ZZ
```

```
 140    CONTINUE
 150 CONTINUE
     GO TO 155
C
C******                                  NO CONVERGENCE AFTER 30 ITERATIONS
C******                                  SET ERROR INDICATOR  TO THE INDEX
C******                                  OF THE CURRENT EIGENVALUE
C
 151 IER=128+IEN
     DO 152 I=1,IEN
        WRL(I)=ZERO
        WRI(I)=ZERO
 152 CONTINUE
     IF (IZ.LT.N) GO TO 155
     DO 154 I=1,N
        DO 153 J=1,N
           Z(I,J)=ZERO
 153    CONTINUE
 154 CONTINUE
 155 CONTINUE
     CALL FERTST (IER,6HFCRALG)
 156 RETURN
C
     END
     SUBROUTINE FRCHSS (A,K,L,N,IA,D)
C
C*****************************************************************************
C      *                                                                     *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:                   *
C      *        1. REDUCES A REAL MATRIX TO UPPER HESSENBERG FORM THRU       *
C      *           ORTHOGONAL TRANSFORMATIONS.                               *
C      *                                                                     *
C******  THIS SUB PROGRAM'S GLOSSARY OF FORTRAN NAMES:                       *
C      *        A          - MATRIX TO BE REDUCED TO HESSENBERG FORM         *
C      *        K,L        - SAME AS IN FDKAM2                               *
C      *        N          - ORDER OF A                                      *
C      *        IA         - ROW DIMENSION OF MATRIX A                       *
C      *        D          - DETAILS OF TRANSFORMATION FOR SUBSEQUENT USE    *
C      *                                                                     *
C*****************************************************************************
C
     DIMENSION A(IA,N), D(N)
     DATA ZERO/0.0/
     LA=L-1
     KP1=K+1
     IF (LA.LT.KP1) GO TO 110
     DO 100 M=KP1,LA
        H=ZERO
        D(M)=ZERO
        SCALE=ZERO
C
C******                                  SCALE COLUMN
C
        DO 101 I=M,L
           SCALE=SCALE+ABS(A(I,M-1))
 101    CONTINUE
        IF (SCALE.EQ.ZERO) GO TO 100
        MP=M+L
C
C******                                  DO 10 I=L,M,-1
```

```
C
      DO 102 II=M,L
         I=MP-II
         D(I)=A(I,M-1)/SCALE
         H=H+D(I)*D(I)
  102 CONTINUE
      G=-SIGN(SQRT(H),D(M))
      H=H-D(M)*G
      D(M)=D(M)-G
      DO 105 J=M,N
         F=ZERO
C
C******                               DO 15 I=L,M,-1
C
         DO 103 II=M,L
            I=MP-II
            F=F+D(I)*A(I,J)
  103    CONTINUE
         F=F/H
         DO 104 I=M,L
            A(I,J)=A(I,J)-F*D(I)
  104    CONTINUE
  105 CONTINUE
      DO 108 I=1,L
         F=ZERO
C
C******                               DO 30 J=L,M,-1
C
         DO 106 JJ=M,L
            J=MP-JJ
            F=F+D(J)*A(I,J)
  106    CONTINUE
         F=F/H
         DO 107 J=M,L
            A(I,J)=A(I,J)-F*D(J)
  107    CONTINUE
  108 CONTINUE
      D(M)=SCALE*D(M)
      A(M,M-1)=SCALE*G
  109 CONTINUE
  110 RETURN
C
      END
```

```
      SUBROUTINE GZOC (N,EV,EVALS,BMAT,CMAT,DMAT,BH,CH,X,Y,NPORT,PR,MP,M
     1S)
C
C***********************************************************************
C      *                                                               *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTIONS:            *
C      *        1. OBTAIN AND STORE COMPLETE INFORMATION ABOUT THE     *
C      *           OPEN-CIRCUIT IMPEDANCE MATRIX IN PARTIAL FRACTION   *
C      *           EXPANSION (PFE) FORM.                               *
C      *        2. CHECK OF JW-AXIS OR REPEATED EIGENVALUES.           *
C      *        3. PRINT ENTRIES OF ZOC, IF REQUESTED.                 *
C      *                                                               *
C******  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINE:               *
C      *        1. GZOCPR                                              *
C      *        2. **** LINEQ4 **** LIBRARY DEPENDENT ROUTINE          *
C      *                                                               *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                 *
C      *  ALL VARIABLE NAMES AND ARRAYS AS DEFINED IN SUB-PROGRAM AMAIN *
C      *                                                               *
C***********************************************************************
C
      COMPLEX EVALS(1),EV(MS,1),X(MS,1),Y(MS,1),SUM,BH(MS,1)
      COMPLEX CH(MP,1)
      INTEGER ER,TYPE,PR
      DIMENSION BMAT(MS,1), CMAT(MP,1), DMAT(MP,1), NPORT(1)
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NDCS,NCPRT
C
C*****OBTAIN THE INVERSE OF THE MODAL MATRIX
C
      DO 104 I=1,N
         DO 102 J=1,N
  102    Y(I,J)=CMPLX(0.00,0.00)
  104 Y(I,I)=CMPLX(1.00,0.00)
C
C***************SYSTEM DEPENDENT ROUTINE FOR FINDING INVERSE OF A COMPX
C*************** MATRIX
C
      CALL LINEQ4 (EV,Y,X,20,N,N,IERR)
      IF (IERR.NE.0) GO TO 122
      DO 103 I=1,N
      DO 103 J=1,N
         U=REAL(EV(I,J))
         V=AIMAG(EV(I,J))
         U1=REAL(X(I,J))
         V1=AIMAG(X(I,J))
         IF (ABS(U).LT.1.00E-15) U=0.0000
         IF (ABS(V).LT.1.00E-15) V=0.0000
         IF (ABS(U1).LT.1.00E-15) U1=0.000
         IF (ABS(V1).LT.1.00E-15) V1=0.000
         EV(I,J)=CMPLX(U,V)
         X(I,J)=CMPLX(U1,V1)
  103 CONTINUE
C
C***** FORM THE TINV*DMAT PRODUCT
C
      DO 110 I=1,N
      DO 110 J=1,NCPRT
         SUM=CMPLX(0.0000,0.0000)
         DO 108 K=1,N
  108    SUM=SUM+X(I,K)*DMAT(K,J)
```

```
         BH(I,J)=SUM
  110 CONTINUE
C
C*****FORM THE PRODUCT C*T
C
      DO 114 I=1,NCPRT
      DO 114 J=1,N
         SUM=CMPLX(0.00,0.00)
         DO 112 K=1,N
  112    SUM=SUM+CMAT(I,K)*EV(K,J)
         CH(I,J)=SUM
  114 CONTINUE
C
C***** CHECK FOR REPEATED OR JW-AXIS EIGENVALUES:
C
      IWARN1=0
      IWARN2=0
      DO 120 I=1,N
         U1=REAL(EVALS(I))
         V1=AIMAG(EVALS(I))
         J=I+1
  116    IF (J.GT.N) GO TO 118
         U2=REAL(EVALS(J))
         V2=AIMAG(EVALS(J))
         AU=ABS(U2-U1)
         AV=ABS(V2-V1)
         IF ((AU.LT.1.000E-08).AND.(AV.LT.1.00E-08)) IWARN1=1
         J=J+1
         GO TO 116
  118    IF (ABS(U1).GT.1.00E-08) GO TO 120
         IWARN2=1
         EVALS(I)=CMPLX(-0.1000,V1)
  120 CONTINUE
      IF (IWARN1.EQ.1) WRITE (6,124)
      IF (IWARN2.EQ.1) WRITE (6,126)
C
C*****WRITE THE INVERSE OF THE NODE ADMITTANCE IN PFE FORM, IF DESIRED
C
      IF (PR.NE.1) RETURN
      CALL GZOCPR (CH,BH,Y,N,NPORT,DMAT,EVALS,MP,MS)
      RETURN
  122 WRITE (6,128)
      RETURN
C
  124 FORMAT (1H0,13H** WARNING **,/1H ,47HREPEATED EIGENVALUES, ANSWERS
     1 MAY BE INACCURATE)
  126 FORMAT (1H0,13H** WARNING **,/1H ,37HJW-AXIS POLE PRESENT HAS BEEN
     1 SHIFTED)
  128 FORMAT (1H0,22HSINGULAR MODAL MATRIX )
C
      END
      SUBROUTINE GZOCPR (CHAT,BHAT,X,NSTU,NPORT,DMAT,EVALS,MP,MS)
C
C***********************************************************************
C******   THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:             *
C     *        1. PRINT THE ENTRIES OF THE OPEN-CIRCUIT IMPEDANCE MATRIX*
C     *           IN PARTIAL FRACTION EXPANSION FORM, IF REQUESTED.     *
C     *                                                                 *
C******   THIS SUB-PROGRAM≠S GLOSSARY OF FORTRAN NAMES:                 *
C     *        ALL VARIABLE NAMES AND ARRAYS AS DEFINED IN SUB-PROGRAM  *
```

```
C     *         ZOC.                                                    *
C     *                                                                 *
C*******************************************************************
C
      COMPLEX CHAT(MP,1),BHAT(MS,1),X(MS,1),EVALS(1)
      DIMENSION NPORT(1), DMAT(MP,1)
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NDCS,NCPRT
      WRITE (6,106)
      DO 104 I=1,NCPRT
      DO 104 J=1,NCPRT
         WRITE (6,108) NPORT(I),NPORT(J)
         DO 102 K=1,NSTV
  102    X(1,K)=CHAT(I,K)*BHAT(K,J)
         WRITE (6,110) (X(1,K),EVALS(K),K=1,NSTV)
         WRITE (6,112) DMAT(I,J)
  104 CONTINUE
      RETURN
C
  106 FORMAT (1H1,29HOPEN CIRCUIT IMPEDANCE MATRIX)
  108 FORMAT (1X,2HZ(,1X,I2,1H,,1X,I2,2H):,/1H ,12X,7HRESIDUE,27X,10HEIG
     1ENVALUE)
  110 FORMAT (1H ,5X,E12.5,2H+J,E12.5,4X,E12.5,2H+J,E12.5)
  112 FORMAT (1H0,9HCONSTANT=,E12.5)
C
      END
      SUBROUTINE HORDR1 (NFREQ,NSTV,BHAT,CHAT,EV,H,N2FREQ,DMT,ZS,ZS1,LOS
     1RC,NZT,NZT1,MIX,PHASE,MP,MS)
C
C*******************************************************************
C     *                                                                 *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:            *
C     *         1. COMPUTE THE FIRST-ORDER TRANSFER FUNCTION AT EACH   *
C     *            POSITIVE AND NEGATIVE INPUT FREQUENCY VALUE.         *
C     *                                                                 *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                 *
C     *         NFREQ     : NUMBER OF POSITIVE INPUT FREQUENCIES       *
C     *         NSTV      : NUMBER OF STATE VARIABLES (CIRCUIT COMPLEXITY*
C     *         H(I,J)    : I-TH PORT FIRST-ORDER TRANSFER FUNCTION VALUE*
C     *                     AT W1(J) FREQUENCY VALUE                    *
C     *         ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN      *
C     *         SUB-PROGRAM AMAIN                                       *
C     *                                                                 *
C*******************************************************************
C
      COMPLEX SUM,S,EV(1),CHAT(MP,1),BHAT(MS,1),H(MP,1),TH
      DIMENSION DMT(MP,1), NPORT(1)
      COMMON /003/ W1(10),AMP(10),TH(10),LUNIT
      DIMENSION PHASE(5)
      COMMON /016/ NCONT(32),JCONT(10)
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NDCS,NOUT
C
C****** FORM POSITIVE AND NEGATIVE FREQUENCY ARRAY FOR ANALYSIS
C
      DO 105 I=1,NFREQ
         K=NFREQ+I
         PHASE(I)=3.141592654*PHASE(I)/180.0000
         TH(I)=CMPLX(0.0000,PHASE(I))
         TH(K)=CMPLX(0.0000,-PHASE(I))
         AMP(K)=AMP(I)
  105 W1(K)=-W1(I)
```

```
C
C***** OBTAIN THE FIRST-ORDER TRANSFER FUNCTION AT EACH FREQUENCY POINT
C
      DO 140 L=1,NFREQ
         IF (LUNIT.EQ.2H HZ) WI=2.00000*3.141592654*W1(L)
         S=CMPLX(0.00,WI)
         DO 135 I=1,NOUT
            SUM=CMPLX(0.00,0.00)
            DO 110 K=1,NSTV
  110       SUM=SUM+CHAT(I,K)*DHAT(K,1)/(S-EV(K))
            DHAT=DMT(I,1)
            GO TO (115,120,125), NZT
  115       H(I,L)=(SUM+CMPLX(DHAT,0.0000))/CMPLX(ZS,0.000)
            GO TO 130
  120       H(I,L)=(SUM+CMPLX(DHAT,0.0000))*ZS*S
            GO TO 130
  125       H(I,L)=(SUM+CMPLX(DHAT,0.0000))/ZS/S
  130       H(I,NFREQ+L)=CONJG(H(I,L))
  135    CONTINUE
C
  140 CONTINUE
C
C***** COMPUTE RESPONSE DUE TO SECOND-GENERATOR, IF PRESENT
C
      IF (MIX.NE.1) GO TO 175
      INP2=NCONT(LOSRC)
      N2FREQ=2*NFREQ
      DO 170 I=1,NOUT
         SUM=CMPLX(0.0000,0.0000)
         DO 145 K=1,NSTV
  145    SUM=SUM+CHAT(I,K)*BHAT(K,INP2)/(S-EV(K))
         DHAT=DMT(I,INP2)
         GO TO (150,155,160), NZT1
  150    H(I,NFREQ)=(SUM+CMPLX(DHAT,0.000))/CMPLX(ZS1,0.0000)
         GO TO 165
  155    H(I,NFREQ)=(SUM+CMPLX(DHAT,0.000))*ZS1*S
         GO TO 165
  160    H(I,NFREQ)=(SUM+CMPLX(DHAT,0.000))/ZS1/S
  165    H(I,N2FREQ)=CONJG(H(I,NFREQ))
  170 CONTINUE
  175 RETURN
C
      END
      SUBROUTINE HORDR2 (NFREQ,NSTV,NNELEM,EV,DHAT,CHAT,H,H2,N2FREQ,W2,N
     1ICS,TEMP,DMT,N2FRPT,FCV,NPOUT,MP,MS)
C
C*************************************************************************
C     *
C*****>   THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:
C     *        1. COMPUTE THE SECOND-ORDER TRANSFER FUNCTION VALUES AT   *
C     *           COMBINATION OF A PAIR OF POSITIVE AND NEGATIVE INPUT   *
C     *           FREQUENCY VALUES.                                      *
C     *                                                                  *
C*****>   THIS SUB-PROGRAM'S GLOSSARY OF FORTRAN NAMES:                  *
C     *        H2(I,J)    : I-TH PORT SECOND-ORDER TRANSFER FUNCTION     *
C     *                     VALUE AT FREQUENCY W2(J)                     *
C     *        W2(I)      : I-TH FREQUENCY VALUE APPEARING IN THE SECOND-*
C     *                     ORDER SPECTRUM                               *
C     *        N2FRPT     : TOTAL NUMBER OF FREQUENCY POINTS APPEARING   *
C     *                     IN THE SECOND-ORDER SPECTRUM                 *
```

```
C      *        FCV(I)     : I-TH FREQUENCY COMBINATION CODE                  *
C      *        SRC2(L)    : SECOND-ORDER CURRENT SOURCE DUE TO THE L-TH      *
C      *                     NONLINEAR ELEMENT                                *
C      *        ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN             *
C      *        SUB-PROGRAM AMAIN                                             *
C      *                                                                      *
C***********************************************************************
C
      INTEGER FCV(1)
      COMPLEX SUM,S,EV(1),EHAT(MS,1),CHAT(MP,1),H(MP,1),H2(MP,1)
      COMPLEX SRC2(25),TEMP(MP,1),SP,S3,S5,TH
      DIMENSION W2(1), ENT(MP,1)
      COMMON /001/ NTYPE(10),AI(10,9)
      COMMON /003/ W1(10),AMP(10),TH(10),LUNIT
      COMMON /016/ NCONT(22),JCONT(10)
      COMMON /ENGS/ NCAP,NIVS,NRES,NIND,NCCS,NCS
      DATA NL/2HNL/
      K=0
      N2FREQ=2*NFREQ
C
C*****INITIALIZE
C
      DO 105 I=1,NICS
  105 SRC2(I)=CMPLX(0.00,0.00)
C
C***** COMPUTE SECOND-ORDER TRANSFER FUNCTIONS AT EACH FREQUENCY COMBI
C
      DO 155 II=1,N2FREQ
      DO 155 JJ=II,N2FREQ
         DUM=W1(II)+W1(JJ)
         K=K+1
         W2(K)=DUM
         FCV(K)=10*II+JJ
         IF (LUNIT.EQ.2H HZ) DUM=2.000000*3.141592654*(W1(II)+W1(JJ))
         S=CMPLX(0.00,DUM)
C
C***** FORM SECOND-ORDER CURRENT SOURCE VECTOR
C
         DO 130 L=1,NNELEM
            L3=3*(L-1)+NPOUT+2
            ICON1=NCONT(L3)
            ICON2=NCONT(L3+1)
            INDEX=JCONT(L)
            GO TO (110,115,120,125), INDEX
C
C***** NONLINEAR CAPACITIVE SOURCE
C
  110       SRC2(L)=H(ICON1,II)*H(ICON2,JJ)*AI(L,2)*S
            GO TO 130
C
C***** NONLINEAR INDUCTIVE SOURCE
C
  115       IF (DUM.EQ.0.00) GO TO 130
            SRC2(L)=H(ICON1,II)*H(ICON2,JJ)*AI(L,2)/S
            GO TO 130
C
C***** NONLINEAR DEPENDENT SOURCE
C
  120       SP=H(ICON1,II)*H(ICON1,JJ)*AI(L,3)
            S3=H(ICON2,II)*H(ICON2,JJ)*AI(L,4)
```

```
         SS=(H(ICON1,II)*H(ICON2,JJ)+H(ICON2,II)*H(ICON1,JJ))*AI(L,5)
     1   /2.00
         SRC2(L)=+(SP+SQ+SS)
         GO TO 130
C
C***** NONLINEAR RESISTIVE SOURCE
C
  125    SRC2(L)=H(ICON1,II)*H(ICON2,JJ)*AI(L,2)
  130 CONTINUE
C
C***** FORM ZOC( S1+S2 )
C
      DO 140 J=1,NCS
      DO 140 M=1,NCS
         SUM=CMPLX(0.00,0.00)
         DO 135 L=1,NSTV
  135    SUM=SUM+CHAT(J,L)*BHAT(L,M)/(S-EV(L))
         DHAT=DMT(J,M)
         TEMP(J,M)=SUM+CMPLX(DHAT,0.0000)
  140 CONTINUE
C
C***** OBTAIN SECOND-ORDER TRANSFER FUNCTIONS
C
      DO 150 J=1,NCS
         SUM=CMPLX(0.00,0.00)
         DO 145 M=1,NNELEM
            M3=3*(M-1)+NPOUT+1
            ICON=NCONT(M3)
  145    SUM=SUM+TEMP(J,ICON)*SRC2(M)
         H2(J,K)=SUM
         IF ((NTYPE(J).EQ.NL).AND.(SUM.EQ.00.00)) H2(J,K)=0.00
  150 CONTINUE
C
  155 CONTINUE
      N2FRPT=K
      RETURN
C
      END
      SUBROUTINE HORDR3 (NFR,NSTV,NNELEM,EV,BHAT,CHAT,H1,H2,N2F,W3,H3,NI
     1CS,TEMP,DMT,KK,FCV,NPOUT,MP,MS)
C
C************************************************************************
C     *                                                                 *
C******   THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:             *
C     *        1. COMPUTE THE THIRD-ORDER TRANSFER FUNCTION VALUES AT   *
C     *           EACH POSITIVE COMBINATION OF THREE POSITIVE AND       *
C     *           NEGATIVE INPUT FREQUENCIES TAKEN AT A TIME.           *
C     *                                                                 *
C******   THIS SUB-PROGRAM≠S GLOSSARY OF FORTRAN NAMES:                 *
C     *        H3(I,J)    : I-TH PORT THIRD-ORDER TRANSFER FUNCTION VALUE*
C     *                     AT FREQUENCY W3(J)                          *
C     *        W3(J)      : J-TH POSITIVE FREQUENCY VALUE APPEARING IN  *
C     *                     THE THIRD-ORDER SPECTRUM                    *
C     *        FCV(J)     : W3(J) FREQUENCY COMBINATION CODE            *
C     *        SRC3(L)    : THIRD-ORDER CURRENT SOURCE DUE TO L-TH      *
C     *                     NONLINEAR ELEMENT                           *
C     *        ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN        *
C     *        SUB-PROGRAM AMAIN                                        *
C     *                                                                 *
C************************************************************************
```

```
C
      INTEGER FCV(1)
      COMPLEX SUM,S,EV(1),EHAT(MS,1),CHAT(MP,1),H1(MP,1),H2(MP,1),SRC3(2
     15),TEMP(MP,1),H3(MP,1),G31,G32,G33,G34,G231,G232,G233,G234,TH
      DIMENSION W3(1), DMT(MP,1)
      COMMON /001/ NTYPE(10),AI(10,9)
      COMMON /003/ W1(10),AMP(10),TH(10),LUNIT
      COMMON /016/ NCONT(32),JCONT(10)
      COMMON /ENDS/ NCAP,NDVS,NRES,NIND,NDCS,NCS
      DATA NL/2HNL/
      KK=0
C
C***** INITIALIZE
C
      DO 105 I=1,NICS
  105 SRC3(I)=CMPLX(0.00,0.00)
C
C***** COMPUTE THIRD-ORDER TRANSFER FUNCTION AT EACH FREQUENCY COMB
C
      DO 155 I=1,NFR
      DO 155 J=I,N2F
      DO 155 K=J,N2F
         DUM=W1(I)+W1(J)+W1(K)
         IF (DUM.LT.0.00) GO TO 155
         KK=KK+1
         W3(KK)=DUM
         FCV(KK)=100*I+10*J+K
         IF (LUNIT.EQ.2H HZ) DUM=2.0000*3.141592654*DUM
         S=CMPLX(0.00,DUM)
         IDUM=(I-1)*N2F-I*(I-1)/2
         I1=IDUM+J
         I2=IDUM+K
         J1=(J-1)*N2F-J*(J-1)/2+K
C
C***** FORM NONLINEAR CURRENT SOURCE VECTOR
C
         DO 130 L=1,NNELEM
            L3=3*(L-1)+NPOUT+2
            IC1=NCONT(L3)
            IC2=NCONT(L3+1)
            G31=H1(IC1,I)*H1(IC1,J)*H1(IC1,K)
            G231=H1(IC1,I)*H2(IC1,J1)+H1(IC1,J)*H2(IC1,I2)+H1(IC1,K)*H2(
     1      IC1,I1)
            G231=2.0000*G231/3.00000
            INDEX=JCONT(L)
            GO TO (110,115,120,125), INDEX
C
C***** NONLINEAR CAPACITIVE SOURCE
C
  110       SRC3(L)=(G31*AI(L,3)+G231*AI(L,2))*S
            GO TO 130
C
C***** NONLINEAR INDUCTIVE SOURCE
C
  115       IF (DUM.EQ.0.00) GO TO 130
            SRC3(L)=(G31*AI(L,3)+G231*AI(L,2))/S
            GO TO 130
C
C***** NONLINEAR DEPENDENT SOURCE
C
```

```
  120       G232=H1(IC2,I)*H2(IC2,J1)+H1(IC2,J)*H2(IC2,I2)+H1(IC2,K)*H2(
     1      IC2,I1)
            G233=H1(IC1,I)*H2(IC2,J1)+H1(IC1,J)*H2(IC2,I2)+H1(IC1,K)*H2(
     1      IC2,I1)
            G234=H1(IC2,I)*H2(IC1,J1)+H1(IC2,J)*H2(IC1,I2)+H1(IC2,K)*H2(
     1      IC1,I1)
            G32=H1(IC2,I)*H1(IC2,J)*H1(IC2,K)*AI(L,7)
            G33=(H1(IC1,I)*H1(IC1,J)*H1(IC2,K)+H1(IC1,J)*H1(IC1,K)*H1(IC
     1      2,I)+H1(IC1,K)*H1(IC1,I)*H1(IC2,J))*AI(L,8)/3.0000
            G34=(H1(IC1,I)*H1(IC2,J)*H1(IC2,K)+H1(IC1,J)*H1(IC2,K)*H1(IC
     1      2,I)+H1(IC1,K)*H1(IC2,I)*H1(IC2,J))*AI(L,9)/3.000
            G231=G231*AI(L,3)
            G232=2.0000*G232*AI(L,4)/3.00000
            SRC3(L)=G231+G232+(G233+G234)*AI(L,5)/3.00+G31*AI(L,6)+G32+G
     1      33+G34
            GO TO 130
C
C***** NONLINEAR RESISTIVE SOURCE
C
  125       SRC3(L)=G31*AI(L,3)+G231*AI(L,2)
  130    CONTINUE
C
C***** FORM ZOC ( S1+S2+S3 )
C
         DO 140 JJ=1,NCS
         DO 140 M=1,NCS
            SUM=CMPLX(0.00,0.00)
            DO 135 L=1,NSTV
  135       SUM=SUM+CHAT(JJ,L)*BHAT(L,M)/(S-EV(L))
            DHAT=DMT(JJ,M)
            TEMP(JJ,M)=SUM+CMPLX(DHAT,0.000)
  140    CONTINUE
         DO 150 JJ=1,NCS
            SUM=CMPLX(0.00,0.00)
            DO 145 M=1,NNELEM
               M3=3*(M-1)+NPOUT+1
               ICON=NCONT(M3)
  145       SUM=SUM+TEMP(JJ,ICON)*SRC3(M)
            H3(JJ,KK)=SUM
            IF ((NTYPE(JJ).EQ.NL).AND.(DUM.EQ.0.00)) H3(JJ,KK)=0.00
  150    CONTINUE
C
  155 CONTINUE
      RETURN
C
      END
      SUBROUTINE IWR1ST (NFREQ,H1,NPORT,IAP,NOUT,MP)
C
C***********************************************************************
C     *                                                                *
C******    THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:           *
C     *         1. PRINT THE FIRST-ORDER TRANSFER FUNCTION AND OUTPUT  *
C     *            VOLTAGE VALUES AT EACH POSITIVE INPUT FREQUENCY VALUE *
C     *            AT THE REQUESTED PORTS.                             *
C     *                                                                *
C******    THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:               *
C     *         H1(I,J)   : UPON ENTRANCE: I-TH PORT FIRST-ORDER TRANSFER*
C     *                     FUNCTION VALUE AT W1(J); UPON EXIT: I-TH PORT*
C     *                     OUTPUT VOLTAGE VALUE AT FREQUENCY W1(J)    *
C     *         IAP       : PRINTING OPTION FLAG VARIABLE              *
```

```
C     *        ALL OTHER VARIABLES NAMES AND ARRAYS AS DEFINED IN          *
C     *        SUB-PROGRAM AMAIN.                                          *
C     *                                                                    *
C***********************************************************************
C
      DIMENSION NPORT(1)
      COMPLEX H1(MP,1),TH
      COMMON /003/ W1(10),AMP(10),TH(10),LUNIT
      COMMON /ENGS/ NCAP,NDVS,NRES,NIND,NDCS,NCS
C
C***** CHECK IF RESPONSE IS TO BE PRINTED AT ALL EXTRACTED PORTS
C
      IF (IAP.EQ.1) GO TO 105
      L=2
      K=NOUT
      GO TO 110
  105 L=1
      K=NCS
C
C***** PRINT THE FIRST-ORDER TRANSFER FUNCTION AND RESPONSE AT EACH
C***** OUTPUT PORT AND FREQUENCY
C
  110 DO 130 I=1,NFREQ
         WRITE (6,135) I,W1(I),LUNIT
         WRITE (6,140)
         WRITE (6,145)
         DO 125 J=L,K
            IOUT=NPORT(J)
            AMAGN=CABS(H1(IOUT,I))
            U=-REAL(H1(IOUT,I))
            V=-AIMAG(H1(IOUT,I))
            H1(IOUT,I)=AMP(I)*H1(IOUT,I)*CEXP(TH(I))
            YMAG=CABS(H1(IOUT,I))
            YU=-REAL(H1(IOUT,I))
            YV=-AIMAG(H1(IOUT,I))
            IF (AMAGN.EQ.0.0000) GO TO 115
            ADB=20.000*ALOG10(AMAGN)
            PHASE=ATAN2(YV,YU)*180.000/3.141592654
            GO TO 120
  115       ADB=-1.00E+30
            PHASE=0.00000
  120       WRITE (6,150) IOUT,U,V,AMAGN,ADB,YU,YV,YMAG,PHASE
  125    CONTINUE
  130 CONTINUE
      RETURN
C
  135 FORMAT (1H0,12HFIRST ORDER:,15X,11HFREQUENCY( ,I1,5H )=  ,E10.3,2X
     1,A3)
  140 FORMAT (1H0,34X,17HTRANSFER FUNCTION,40X,14HOUTPUT VOLTAGE)
  145 FORMAT (//1X,9H    PORT   ,8X,4HREAL,8X,9HIMAGINARY,6X,9HMAGNITUDE,6
     1X,9H20LOG MAG,9X,4HREAL,8X,9HIMAGINARY,6X,9HMAGNITUDE,6X,5HPHASE,/
     2,5X,2HNO,114X,3HDEG,/,1H ,3X,4(1H.),6X,57(1H.),3X,53(1H.))
  150 FORMAT (1H ,4X,I2,4X,7(3X,E12.5),3X,F7.2)
C
      END
      SUBROUTINE IWR2ND (NFREQ,N2FRPT,H2,NPORT,I2FC,W2,IAP,NOUT,MP)
C
C***********************************************************************
C     *                                                                    *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:                 *
```

```
C       *          1. PRINT THE SECOND-ORDER TRANSFER FUNCTION AND OUTPUT   *
C       *             VOLTAGE VALUES AT EACH OF THE NON-NEGATIVE FREQUENCY  *
C       *             VALUES AT THE REQUESTED PORTS.                        *
C       *                                                                   *
C******* THIS SUB-PROGRAM#S GLOSSARY OF FORTRAN NAMES:                       *
C       *         N2FRPT     : TOTAL NUMBER OF FREQUENCY COMPONENTS IN THE  *
C       *                        SECOND-ORDER RESPONSE                      *
C       *         H2(I,J)    : UPON ENTRANCE: I-TH PORT SECOND-ORDER TRANSFE*
C       *                        FUNCTION VALUE AT W2(J); UPON EXIT: I-TH PORT*
C       *                        OUTPUT VOLTAGE VALUE AT FREQUENCY W2(J)    *
C       *         I2FC(J)    : W2(J) FREQUENCY COMBINATION CODE             *
C       *         ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN         *
C       *         SUB-PROGRAM AMAIN.                                        *
C       *                                                                   *
C********************************************************************************
C
      DIMENSION NPORT(1), I2FC(1), W2(1)
      COMPLEX H2(MP,1),TH
      COMMON /003/ W1(10),AMP(10),TH(10),LUNIT
      COMMON /ENDS/ NCAP,NDUS,NRES,NIND,NCCS,NCS
C
C*****CHECK IF RESPONSE IS TO BE PRINTED AT ALL EXTRACTED PORTS
C
      IF (IAP.EQ.1) GO TO 105
      L=2
      K=NOUT
      GO TO 110
  105 L=1
      K=NCS
C
C*****PRINT SECOND-ORDER TRANSFER FUNCTION AND RESPONSE AT EACH OUTPUT
C*****PORT AND POSITIVE FREQUENCY
C
  110 DO 130 I=1,N2FRPT
         IF (W2(I).LT.0.0D0) GO TO 130
C
C***** DECIPHER FREQUENCY COMBINATION
C
         ICOMB=I2FC(I)
         II=ICOMB/10
         JJ=ICOMB-10*II
         LL=JJ
         IF (LL.GT.NFREQ) LL=NFREQ-LL
         WRITE (6,135) II,LL,W2(I),LUNIT
         WRITE (6,140)
         WRITE (6,145)
         DO 125 J=L,K
            IOUT=NPORT(J)
            AMAGN=CABS(H2(IOUT,I))
            U=REAL(H2(IOUT,I))
            V=AIMAG(H2(IOUT,I))
            H2(IOUT,I)=AMP(II)*AMP(JJ)*H2(IOUT,I)*CEXP(TH(II))*CEXP(TH(J
     1      J))
            IF (II.EQ.JJ) H2(IOUT,I)=H2(IOUT,I)/CMPLX(2.0D0,0.0D0)
            YMAG=CABS(H2(IOUT,I))
            YU=REAL(H2(IOUT,I))
            YV=AIMAG(H2(IOUT,I))
            IF (AMAGN.EQ.0.0000) GO TO 115
            ADB=20.0000*ALOG10(AMAGN)
            PHASE=ATAN2(YV,YU)*180.000/3.14159265
```

```
         GO TO 120
  115    ADB=-1.000E+30
         PHASE=0.0000
  120    WRITE (6,150) IOUT,U,V,AMAGN,ADB,YU,YV,YMAG,PHASE
  125    CONTINUE
  130 CONTINUE
      RETURN
C
  135 FORMAT (1H0,13HSECOND ORDER:,15X,11HFREQUENCY( ,I1,1H,,I2,5H )=  ,
     1E10.3,2X,A3)
  140 FORMAT (1H0,34X,17HTRANSFER FUNCTION,40X,14HOUTPUT VOLTAGE)
  145 FORMAT (//1X,8H   PORT  ,6X,4HREAL,8X,9HIMAGINARY,6X,9HMAGNITUDE,6
     1X,9H20*LOG MAG,9X,4HREAL,8X,9HIMAGINARY,6X,9HMAGNITUDE,6X,5HPHASE,/
     2,5X,2HNO,114X,5HDEG,/,1H ,3X,4(1H.),6X,57(1H.),3X,53(1H.))
  150 FORMAT (1H ,4X,I2,4X,7(3X,E12.5),3X,F7.2)
C
      END
      SUBROUTINE IWRORD (NFREQ,NSFRPT,H3,NPORT,ISFC,W3,IAP,NOUT,MP)
C
C*****************************************************************
C     *                                                          *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:        *
C     *        1. PRINT THE THIRD-ORDER TRANSFER FUNCTION AND OUTPUT *
C     *           VOLTAGE VALUES AT EACH OF THE NON-NEGATIVE FREQUENCY *
C     *           VALUES AT THE REQUESTED OUTPUT PORTS.             *
C     *                                                          *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:           *
C     *        NSFRPT     : TOTAL NUMBER OF NON-NEGATIVE FREQUENCY  *
C     *                     COMPONENTS IN THE THIRD-ORDER SPECTRUM  *
C     *        H3(I,J)    : UPON ENTRANCE: I-TH PORT THIRD-ORDER TRANSFER*
C     *                     FUNCTION VALUE AT W3(J); UPON EXIT: I-TH PORT*
C     *                     THIRD-ORDER OUTPUT VOLTAGE AT FREQUENCY W3(J)*
C     *        ISFC(J)    : W3(J) FREQUENCY COMBINATION CODE        *
C     *        ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN    *
C     *        SUB-PROGRAM AMAIN                                    *
C     *                                                          *
C*****************************************************************
C
      DIMENSION NPORT(1), ISFC(1), W3(1)
      COMPLEX H3(MP,1),TH
      COMMON /CO3/ W1(10),AMP(10),TH(10),LUNIT
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NBCS,NCS
C
C***** CHECK IF RESPONSE IS TO BE PRINTED FOR ALL EXTRACTED PORTS
C
      IF (IAP.EQ.1) GO TO 105
      K=2
      L=NOUT
      GO TO 110
  105 K=1
      L=NCS
C
C***** PRINT THIRD-ORDER TRANSFER FUNCTION AND RESPONSE AT EACH OUTPUT
C***** PORT AND POSITIVE FREQUENCY POINT
C
  110 DO 130 I=1,NSFRPT
C
C***** DECIPHER FREQUENCY COMBINATION
C
         ICOMB=ISFC(I)
```

```
      II=ICOMB/100                                                      IW3
      JJ=(ICOMB-100*II)/10                                              IW3
      KK=ICOMB-100*II-10*JJ                                             IW3
      J1=JJ                                                             IW3
      K1=KK                                                             IW3
      IF (J1.GT.NFREQ) J1=NFREQ-J1                                      IW3
      IF (K1.GT.NFREQ) K1=NFREQ-K1                                      IW3
      WRITE (6,135) II,J1,K1,W3(I),LUNIT                                IW3
      WRITE (6,140)                                                     IW3
      WRITE (6,145)                                                     IW3
      DO 125 J=K,L                                                      IW3
         IOUT=NPORT(J)                                                  IW3
         AMAGN=CABS(H3(IOUT,I))                                         IW3
         U=REAL(H3(IOUT,I))                                             IW3
         V=AIMAG(H3(IOUT,I))                                            IW3
         DIV=CMPLX(1.00,0.00)                                           IW3
         IF ((II.EQ.JJ).OR.(JJ.EQ.KK)) DIV=CMPLX(2.000,00.00)           IW3
         IF ((II.EQ.JJ).AND.(JJ.EQ.KK)) DIV=CMPLX(6.00,0.00)            IW3
         H3(IOUT,I)=1.500*AMP(II)*AMP(JJ)*AMP(KK)*H3(IOUT,I)/DIV        IW3
         H3(IOUT,I)=H3(IOUT,I)*CEXP(TH(II))*CEXP(TH(JJ))*CEXP(TH(KK))IW3
         YMAG=CABS(H3(IOUT,I))                                          IW3
         YU=REAL(H3(IOUT,I))                                            IW3
         YV=AIMAG(H3(IOUT,I))                                           IW3
         IF (AMAGN.EQ.0.0000) GO TO 115                                 IW3
         ADB=20.000*ALOG10(AMAGN)                                       IW3
         PHASE=ATAN2(YV,YU)*180.00/3.141592654                          IW3
         GO TO 120                                                      IW3
  115    ADB=-1.000E+30                                                 IW3
         PHASE=0.000                                                    IW3
  120    WRITE (6,150) IOUT,U,V,AMAGN,ADB,YU,YV,YMAG,PHASE              IW3
  125 CONTINUE                                                          IW3
  130 CONTINUE                                                          IW3
      RETURN                                                            IW3
C                                                                       IW3
  135 FORMAT (1H0,12HTHIRD ORDER:,15X,11HFREQUENCY( ,I1,1H,,I2,1H,,I2,5HIW3
     1 )=   ,E10.3,2X,A3)                                               IW3
  140 FORMAT (1H0,34X,17HTRANSFER FUNCTION,40X,14HOUTPUT VOLTAGE)       IW3
  145 FORMAT (//1X,9H   PORT   ,8X,4HREAL,8X,9HIMAGINARY,6X,9HMAGNITUDE,6IW3
     1X,9H20LOG MAG,9X,4HREAL,8X,9HIMAGINARY,6X,9HMAGNITUDE,6X,5HPHASE,/IW3
     2,5X,2HNO,114X,3HDEG,/,1H ,3X,4(1H.),6X,57(1H.),3X,53(1H.))        IW3
  150 FORMAT (1H ,4X,I2,4X,7(3X,E12.5),3X,F7.2)                         IW3
C                                                                       IW3
      END                                                               IW3
      SUBROUTINE JSPCTM (Y1,Y2,Y3,NFREQ,N2FRPT,N3FRPT,FR,Y,MOUT,W2,W3,IPJSP
     1T,YLG,MP)                                                         JSP
C                                                                       JSP
C**********************************************************************JSP
C     *                                                                *JSP
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTIONS:            *JSP
C     *       1. PERFORM AN "HISTOGRAM" ANALYSIS OF ALL THE OUTPUT     *JSP
C     *          FREQUENCY COMPONENTS AND COMBINE THE REPEATED ONES.   *JSP
C     *       2. PRINT AND PLOT THE COMPLETE OUTPUT SPECTRUM.          *JSP
C     *                                                                *JSP
C******  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINES:              *JSP
C     *       1. JPLTSF                                                *JSP
C     *                                                                *JSP
C******  THIS SUB-PROGRAM'S GLOSSARY OF FORTRAN NAMES:                 *JSP
C     *        Y1(I,J)   : I-TH PORT FIRST-ORDER RESPONSE AT W1(J)     *JSP
C     *        Y2(I,J)   : I-TH PORT SECOND-ORDER RESPONSE AT W2(J)    *JSP
C     *        Y3(I,J)   : I-TH PORT THIRD-ORDER RESPONSE AT W3(J)     *JSP
```

```
C      *        NFREQ     : TOTAL NUMBER OF INPUT FREQUENCIES                  *
C      *        N2FRPT    : TOTAL NUMBER OF POSITIVE AND NEGATIVE              *
C      *                    FREQUENCIES IN THE SECOND-ORDER RESPONSE           *
C      *        N3FRPT    : TOTAL NUMBER OF NON-NEGATIVE FREQUENCIES           *
C      *                    IN THE THIRD-ORDER RESPONSE                        *
C      *        FR        : VALUES OF DISTINCT FREQUENCIES IN THE OUTPUT       *
C      *                    SPECTRUM                                           *
C      *        Y(I)      : OUTPUT PORT VOLTAGE AT FREQUENCY FR(I)             *
C      *        MOUT      : OUTPUT PORT INDEX                                  *
C      *        YLG(I)    : LOG OF THE OUTPUT VOLTAGE AT FREQUENCY FR(I)       *
C      *                                                                       *
C***********************************************************************************
C
      COMPLEX Y1(MP,1),Y2(MP,1),Y3(MP,1),Y(1),TH
      DIMENSION FR(1), W2(1), W3(1), IPT(1), YLG(1), IFRUNT(2)
      COMMON /003/ W1(10),AMP(10),TH(10),LUNIT
      COMMON /016/ NCONT(22),JCONT(10)
      DATA IFRUNT(1),IFRUNT(2)/7HRAD/SEC,7H HERTZ /
      JOUT=NCONT(MOUT)
      ICON=2
C
C***** PACK THE VARIOUS ORDER RESPONSES FOR THE REQUESTED OUTPUT
C***** PORT INTO AN ARRAY FOR ~HISTOGRAM~ ANALYSIS
C
      DO 105 I=1,NFREQ
         FR(I)=W1(I)
  105 Y(I)=Y1(JOUT,I)
      KOUNT=NFREQ
C
C                 SECOND-ORDER RESPONSE
C
      DO 110 I=1,N2FRPT
         IF (W2(I).LT.0.00) GO TO 110
         KOUNT=KOUNT+1
         Y(KOUNT)=Y2(JOUT,I)
         FR(KOUNT)=W2(I)
  110 CONTINUE
C
C                 THIRD-ORDER RESPONSE
C
      DO 115 I=1,N3FRPT
         KOUNT=KOUNT+1
         Y(KOUNT)=Y3(JOUT,I)
         FR(KOUNT)=W3(I)
  115 CONTINUE
C
C***** INITIALIZE
C
      DO 120 I=1,KOUNT
  120 IPT(I)=I
C
C***** PERFORM ~HISTOGRAM~ ANALYSIS
C
      NFPT1=KOUNT-1
      DO 135 I=1,NFPT1
         IPTVAL=IPT(I)
         IF (IPTVAL.LT.I) GO TO 135
         FFREQ=FR(I)
         I1=I+1
         DO 130 J=I1,KOUNT
```

```
            IF (FR(J).EQ.PFREQ) GO TO 125
            GO TO 130
  125       Y(I)=Y(I)+Y(J)
            IPT(J)=IPTVAL
  130     CONTINUE
  135 CONTINUE
C
C***** PRINT COMPLETE OUTPUT SPECTRUM
C
      IF (LUNIT.EQ.3HRAD) ICON=1
      WRITE (6,155) JOUT
      WRITE (6,160) IFRUNT(ICON)
      DO 150 I=1,KOUNT
         IPTVAL=IPT(I)
         IF (IPTVAL.LT.I) GO TO 150
         AMAGN=CABS(Y(I))
         U=REAL(Y(I))
         V=AIMAG(Y(I))
         IF (AMAGN.EQ.0.0D0) GO TO 140
         YLG(I)=ALOG10(AMAGN)
         PHASE=ATAN2(V,U)*180.0D0/3.141592654
         GO TO 145
  140    PHASE=0.0D0
         YLG(I)=-1.0D0E+30
  145    WRITE (6,165) FR(I),U,V,AMAGN,PHASE
  150 CONTINUE
C
C***** PLOT THE OUTPUT SPECTRUM
C
      WRITE (6,170)
      CALL JPLTSP (FR,YLG,KOUNT,23,23HLOG OF OUTPUT MAGNITUDE)
      WRITE (6,175) IFRUNT(ICON)
      RETURN
C
  155 FORMAT (1H1,//,1X,47HSINUSOIDAL STEADY-STATE OUTPUT RESPONSE AT PO
     1RT,2X,I2,/,1H ,47(1H.))
  160 FORMAT (//7X,9HFREQUENCY,11X,4HREAL,12X,9HIMAGINARY,8X,9HMAGNITUDE
     1,12X,5HPHASE,/8X,A7,67X,3HDEG,/1H ,6X,82(1H.))
  165 FORMAT (1H ,5X,E12.2,4X,E12.3,7X,E12.3,5X,E12.3,7X,E12.3)
  170 FORMAT (1H1,45X,31HRESPONSE MAGNITUDE VS FREQUENCY/)
  175 FORMAT (1H0,55X,11HFREQUENCY (,A7,1H))
C
      END
      SUBROUTINE JPLTSP (XX,YY,NDATA,NB,LABEL2)
C
C***********************************************************************
C     *                                                                *
C******  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:             *
C     *       1. PLOT THE COMPLETE OUTPUT SPECTRUM.                    *
C     *                                                                *
C******  THIS SUB-PROGRAM USES THE FOLLOWING SUBROUTINES:              *
C     *       1. JSEP                                                  *
C     *       2. JLCOMP,JPUT (FUNCTION SUB-PROGRAMS)                   *
C     *                                                                *
C******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                 *
C     *       XX          : X COORDINATE OF DATA (FREQUENCY)           *
C     *       YY          : Y COORDINATE OF DATA (LOG MAGNITUDE OF OUTPUT*
C     *       NB          : NUMBER OF CHARACTERS IN Y AXIS LABEL       *
C     *       LABEL2      : Y AXIS TITLE IN HOLLERITH FORMAT           *
C     *       NDATA       : NUMBER OF DATA POINTS                      *
```

```
C      *                                                                 *JPT
C*************************************************************************JPT
C                                                                         JPT
       REAL JPUT                                                          JPT
       DIMENSION SYMBOL(4), AMASK(10), XSCALE(12), YSCALE(51), PLOT(51,10JPT
     1)                                                                   JPT
       DIMENSION XX(1), YY(1)                                             JPT
       DIMENSION LABEL2(6), LB(60)                                        JPT
       DATA (SYMBOL(I),I=1,4)/10H**********,10H0000000000,10H$$$$$$$$$$,1JPT
     10HXXXXXXXXXX/,(AMASK(I),I=1,10)/77000000000000000003,77000000000JPT
     2000000B,7700000000000003,77000000000003,770000000000B,770000000JPT
     3B,77000000B,770000B,7700B,77B/,BLANK/55555555555555555555B/,DASH/1JPT
     40H----------/,UPLINE/10HIIIIIIIIII/,PLUS/10H++++++++++/             JPT
C                                                                         JPT
C      GENERATE GRAPH LABELS                                              JPT
C                                                                         JPT
       N2=N3                                                              JPT
       CALL JSEP (6,60,N2,LABEL2,LB)                                      JPT
C                                                                         JPT
C      ZERO GRAPH ARRAY TO ALL BLANKS                                     JPT
C                                                                         JPT
       DO 105 I=1,51                                                      JPT
       DO 105 J=1,10                                                      JPT
   105 PLOT(I,J)=BLANK                                                    JPT
C                                                                         JPT
C      FIND DATA  MAXIMUM AND MINIMUMS                                    JPT
C                                                                         JPT
       XMAX=XX(1)                                                         JPT
       XMIN=XMAX                                                          JPT
       YMAX=YY(1)                                                         JPT
       YMIN=YMAX                                                          JPT
       IF (NDATA.LE.0) GO TO 115                                          JPT
       DO 110 J=1,NDATA                                                   JPT
          XMAX=AMAX1(XMAX,XX(J))                                          JPT
          XMIN=AMIN1(XMIN,XX(J))                                          JPT
          YMAX=AMAX1(YMAX,YY(J))                                          JPT
   110 YMIN=AMIN1(YMIN,YY(J))                                             JPT
   115 CONTINUE                                                           JPT
C                                                                         JPT
C      DETERMINE X AND Y INCREMENTS                                       JPT
C                                                                         JPT
       XSC=100./(XMAX-XMIN)                                               JPT
       YSC=50.0/(YMAX-YMIN)                                               JPT
C                                                                         JPT
C      CONSTRUCT HORIZONTAL REFERENCE LINES                               JPT
C                                                                         JPT
       DO 125 I=1,51,10                                                   JPT
          IF (I.LT.2) GO TO 125                                           JPT
          DO 120 J=1,10                                                   JPT
   120    PLOT(I,J)=DASH                                                  JPT
   125 CONTINUE                                                           JPT
C                                                                         JPT
C      CONSTRUCT VERTICAL REFERENCE LINE                                  JPT
C                                                                         JPT
       JPOS=10                                                            JPT
       JWORD=10                                                           JPT
       DO 130 I=1,51                                                      JPT
          SYM=UPLINE                                                      JPT
          TEST=(PLOT(I,JWORD).AND.AMASK(JPOS))                            JPT
          TDASH=(DASH.AND.AMASK(JPOS))                                    JPT
```

```
      IF (JLCOMP(TEST,TDASH).EQ.0) SYM=PLUS
  130 PLOT(I,JWORD)=JPUT(AMASK,I,JWORD,JPOS,SYM,51,PLOT)
C
C     DETERMINE X,Y LOCATION OF DATA POINTS ON GRAPH
C
      NDF=NDATA
      DO 145 J=1,NDF
         JYKK=((YY(J)-YMIN)*YSC+1.5)
         JX=((XX(J)-XMIN)*XSC+0.5)
         JWORD=(JX/10)+1
         JPOS=MOD(JX,10)+1
      DO 145 JY=1,JYKK
         TEST=(PLOT(JY,JWORD).AND.AMASK(JPOS))
         TUP=(UPLINE.AND.AMASK(JPOS))
         TBLANK=(BLANK.AND.AMASK(JPOS))
         TDASH=(DASH.AND.AMASK(JPOS))
         TPLUS=(PLUS.AND.AMASK(JPOS))
         IF (JLCOMP(TBLANK,TEST).EQ.0) GO TO 135
         IF (JLCOMP(TDASH,TEST).EQ.0) GO TO 135
         IF (JLCOMP(TPLUS,TEST).EQ.0) GO TO 135
         IF (JLCOMP(TUP,TEST).NE.0) GO TO 140
C
C     INSERT SYMBOL FOR DATA POINT
C
  135    SYM=SYMBOL(1)
         GO TO 145
C
C     IF MULTIPLE DATA POINTS IN  SAME PLOT LOCATION  USE = SIGN
C    ****NOTE**** = SIGN IS INHIBITED  FOR BAR GRAPH FORM OF OUTPUT
C
  140    SYM=SYMBOL(1)
  145 PLOT(JY,JWORD)=JPUT(AMASK,JY,JWORD,JPOS,SYM,51,PLOT)
C
C     GENERATE  X AND Y SCALES
C
      DO 150 I=1,51
  150 YSCALE(I)=FLOAT(I-1)/YSC+YMIN
      DO 155 I=1,11
         JANE=12-I
         XLOW=XMIN-1.0/XSC
         XINC=100.0/(XMAX-XLOW)
         XUL1=(XMAX-FLOAT(10*I-10)/XINC)
         DMAG=ABS(XUL1)
         IF (DMAG.LE.0.10) XUL1=0.0
  155 XSCALE(JANE)=XUL1
      PRINT 190
      DO 175 I=1,51
         JC=52-I
         IT=11
  160    IT=IT-1
         IF (IT.EQ.1) GO TO 165
         IF (PLOT(JC,IT).EQ.BLANK) GO TO 160
  165    IF (N2.EQ.0) GO TO 170
         PRINT 185, LB(I),YSCALE(JC),(PLOT(JC,J),J=1,IT)
         GO TO 175
  170    PRINT 190, YSCALE(JC),(PLOT(JC,J),J=1,IT)
  175 CONTINUE
      PRINT 190
      PRINT 195
      PRINT 200, (XSCALE(I),I=1,11,2),(XSCALE(I),I=2,10,2)
```

```
      RETURN
C
  180 FORMAT (13X,1HI,20(5H....I))
  185 FORMAT (1X,A1,1X,E9.2,1X,1H+,10A10,A5)
  190 FORMAT (2X,E10.3,1X,1H+,10A10,A5)
  195 FORMAT (4X,11(9X,1HU))
  200 FORMAT (4X,E10.3,3X,5(10X,E10.3)/7X,5(10X,E10.3))
C
      END
      REAL FUNCTION JPUT(AMASK,J,JWORD,JPOS,SYM,NPDM,PLOT)
C
C     JPUT ARRANGES DATA POINTS FOR PLOTTING
C
      DIMENSION AMASK(10), PLOT(NPDM,10)
      REAL JPUT
      JPUT=(PLOT(J,JWORD).AND..NOT.AMASK(JPOS)).OR.(AMASK(JPOS).AND.SYM)
      RETURN
C
      END
      FUNCTION JLCOMP(I,K)
C
C     JLCOMP ARRANGES DATA POINTS FOR PLOTTING
C
      IF (I.GE.0.AND.K.LT.0) GO TO 105
      IF (I.LT.0.AND.K.GE.0) GO TO 115
      IF (I-K) 115,110,105
  105 JLCOMP=1
      RETURN
  110 JLCOMP=0
      RETURN
  115 JLCOMP=-1
      RETURN
C
      END
      SUBROUTINE JSEP (ID1,ID2,M,LAB,LA)
C
C*****JSEP SEPARATES THE ALPHABETS IN THE Y-AXIS LABEL FOR VERTICAL
C***** DISPLAY
C
      DIMENSION LAB(ID1), LA(ID2)
      DATA LANK/10H          /
      IF (M.LE.0) GO TO 120
      DO 105 I=1,ID2
  105 LA(I)=LANK
      LIM=(M-1)/10+1
      DO 115 I1=1,LIM
         N=11*I1-I1+1
         LABEL=LAB(I1)
         K=LABEL
      DO 115 I2=1,10
         JJ=N-I2
         IF (JJ.GT.M) GO TO 110
         K=MOD(K,64)
C
C******  BOTH ISHFTLA AND ISHFTRA ARE SYSTEM DEPENDENT ROUTINES ********
C
         K=ISHFTLA(K,54)
         LA(JJ)=K
  110    LABEL=ISHFTRA(LABEL,6)
  115 K=LABEL
```

```
  120 RETURN
C
      END
      SUBROUTINE KFRINC (INTYP,NFREQ)
C
C***********************************************************************
C     *                                                                *
C*****     THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:            *
C     *         1. COMPUTE THE FREQUENCY INCREMENTS FOR FREQUENCY SWEEP  *
C     *            CAPABILITY.                                          *
C     *                                                                *
C*****     THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                *
C     *         INTYP       : TYPE OF FREQUENCY INCREMENTS (LIN OR LOG) *
C     *         NFREQ       : NUMBER OF INPUT FREQUENCIES (LE. 5)        *
C     *         ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN        *
C     *         SUB-PROGRAM AMAIN.                                      *
C     *                                                                *
C***********************************************************************
C
      COMPLEX PHASE
      COMMON /003/ FRQ(10),AMP(10),PHASE(10),LUNIT
      COMMON /004/ NSTPS(5),FRINC(5),HFR(5)
C .
C***** CHECK IF LINEAR OR LOG INCREMENT IS DESIRED
C
      IF (INTYP.EQ.3HLIN) GO TO 110
C
C***** LOG FREQUENCY INCREMENTS
C
      DO 105 I=1,NFREQ
         STPS=FLOAT(NSTPS(I)-1)
  105 FRINC(I)=(HFR(I)/FRQ(I))**(1.000/(STPS-1.000))
      RETURN
C
C***** LINEAR FREQUENCY INCREMENTS
C
  110 DO 115 I=1,NFREQ
         STPS=FLOAT(NSTPS(I)-1)
  115 FRINC(I)=(HFR(I)-FRQ(I))/STPS
      RETURN
C
      END
      SUBROUTINE KFRULS (I,NFREQ,IFLAG)
C
C***********************************************************************
C     *                                                                *
C*****     THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTION:            *
C     *         1. COMPUTE THE INCREMENTED FREQUENCY VALUES FOR THE NEXT *
C     *            ANALYSIS.                                            *
C     *                                                                *
C*****     THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                *
C     *         I           : NUMBER OF INCREMENTS (ANALYSIS) CARRIED OUT *
C     *                       THUS FAR                                  *
C     *         NFREQ       : NUMBER OF INPUT FREQUENCIES               *
C     *         IFLAG       : (=0): NO FREQUENCY VALUES CHANGED         *
C     *                       (=1): FREQUENCY VALUE(S) CHANGED          *
C     *         ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN        *
C     *         SUB-PROGRAM AMAIN.                                      *
C     *                                                                *
************************************************************************
```

```
C
      COMPLEX PHASE
      COMMON /003/ FR(10),AMP(10),PHASE(10),LUNIT
      COMMON /004/ NSTPS(5),FRINC(5),HFR(5)
      IFLAG=0
C
C***** INCREMENT THE INPUT FREQUENCIES
C
      DO 105 J=1,NFREQ
         IF (I.GT.NSTPS(J)) GO TO 105
         FR(J)=FR(J)+FRINC(J)
         IFLAG=1
  105 CONTINUE
      IF (IFLAG.EQ.0) RETURN
      I=I+1
C
C***** PRINT THE NEW FRECUENCY VALUES
C
      WRITE (6,115) LUNIT
      DO 110 J=1,NFREQ
  110 WRITE (6,120) J,FR(J)
      RETURN
C
  115 FORMAT (1H1,18HINPUT FREQUENCIES:,/1H ,9HFREQUENCY,5X,6HVALUE(,A3,
     11H))
  120 FORMAT (1H0,4X,I1,9X,E10.3)
C
      END
      SUBROUTINE LTRANS (NEG,N1,NADD,KK,NLBN,BR,NFROM,NTO,TYPE,ICONT,VAL
     1UE,NNODE,KEY)
C
C***********************************************************************
C       *  THIS SUB-PROGRAM PERFORMS THE FOLLOWING FUNCTIONS:          *
C       *        1. READ THE BIPOLAR TRANSISTOR PARAMETERS SPECIFIED BY *
C       *           THE USER.                                           *
C       *        2. CALCULATE THE COEFFICIENTS OF THE NONLINEAR ELEMENTS *
C       *           PRESENT IN THE EQUIVALENT TRANSISTOR MODEL.         *
C       *        3. FORM TOPOLOGY DESCRIPTION ARRAYS BASED ON THE       *
C       *           EQUIVALENT REPRESENTATION.                          *
C       *                                                               *
C*******  THIS SUB-PROGRAM=S GLOSSARY OF FORTRAN NAMES:                *
C       *        NEG       : USER SPECIFIED ELEMENT(DEVICE) NUMBER      *
C       *        N1        : NODE NUMBER FOR THE BASE TERMINAL          *
C       *        NADD      : CURRENT HIGHEST BRANCH NUMBER IN THE LINEAR *
C       *                    NETWORK                                    *
C       *        KK        : UPON ENTRANCE: CURRENT NUMBER OF NONLINEAR *
C       *                    ELEMENTS; UPON EXIT: NUMBER OF NONLINEAR   *
C       *                    ELEMENTS AFTER INCLUSION OF TRANSISTOR NON- *
C       *                    LINEAR ELEMENTS                            *
C       *        ALL OTHER VARIABLE NAMES AND ARRAYS AS DEFINED IN      *
C       *        SUB-PROGRAM AMAIN.                                     *
C       *                                                               *
C***********************************************************************
C
      INTEGER BR,TYPE,R,C
      REAL IE,IC,ICMAX,M0,M1,M2,M3,N,K,MU
      DIMENSION BR(1), NFROM(1), NTO(1), TYPE(1), VALUE(1), KEY(1), ICON
     1T(1), NLBN(1)
      COMMON /001/ NTYPE(10),A(10,9)
      COMMON /016/ NCONT(32),JCONT(10)
```

```
      COMMON /ENOS/ NCAP,NDVS,NRES,NIND,NDCS,NCS
      DATA R,C,NR,NC,ND/2H R,2H C,2HNR,2HNC,2HND/
C
C***** NODE NUMBERS FOR EMITTER,COLLECTOR,AND INTERNAL JUNCTION
C
      NE=N1+3
      NCJ=N1+2
      NJ=N1+1
C
C***** READ TRANSISTOR PARAMETERS
C
      READ (5,120) N,VCB,VCBO,MU,IC,ICMAX,AP,HFEMAX
      READ (5,120) K,REF,CJE,CP2,RB,RC,C1,C3
C
C***** EMITTER RESISTIVE NONLINEARITY
C
      NADD=NADD+1
      BR(NADD)=NADD
      NLBN(KK)=NADD
      NFROM(NADD)=NJ
      NTO(NADD)=NE
      TYPE(NADD)=NR
      HFE=HFEMAX/(1.00+AP*((ALOG10(IC/ICMAX))**2))
      IE=IC*(1.00+1.00/HFE)
      G1=37.5*IE
      A(KK,1)=G1
      A(KK,2)=G1**2/IE/2.00000
      A(KK,3)=G1**3/IE**2/6.000000
C
C***** COLLECTOR DEPENDENT NONLINEARITY
C
      M0=1.00/(1.00-(VCB/VCBO)**N)
      M1=N*VCB**(N-1)*M0**2/VCBO**N
      M2=(N-1.0000)*M1/VCB/2.00+M1**2/M0
      DUM1=2.00*M2*((N-1.0000)/2.00/VCB+2.0*M1/M0)/3.0000
      DUM2=M1*((N-1.0000)/2.00/VCB**2+(M1/M0)**2)/3.000
      M3=DUM1-DUM2
      SM1=IC*M1/M0
      SM2=IC*M2/M0
      SM3=IC*M3/M0
      DUM2=ALOG10(2.718281828)*2.000*AP
      DUM1=ALOG10(IC/ICMAX)
      A1=HFEMAX/(HFEMAX+1.00+AP*DUM1**2+DUM1*DUM2)
      A2=-A1**3*DUM2*(DUM1+ALOG10(2.718281818))/2.0/IC/HFEMAX
      A3=(A1/6.00)*(-2.00*A2/IC+12.00*(A2/A1)**2-A1**3*DUM2**2/2.00/AP/I
     1C**2/HFEMAX)
      JJ=KK+1
      NADD=NADD+1
      NLBN(JJ)=NADD
      BR(NADD)=NADD
      NFROM(NADD)=NCJ
      NTO(NADD)=NJ
      TYPE(NADD)=ND
      ICONT(NADD)=NADD+2
      JCONT(JJ)=NADD+1
      A(JJ,1)=A1*M0*A(KK,1)
      A(JJ,2)=SM1
      A(JJ,3)=A2*M0*A(KK,1)**2+A1*M0*A(KK,2)
      A(JJ,4)=SM2
      A(JJ,5)=A1*M1*A(KK,1)
```

```
      A(JJ,6)=A3*M0*A(KK,1)**3+A1*M0*A(KK,3)+2.0*A2*M0*A(KK,1)*A(KK,2)
      A(JJ,7)=SM3
      A(JJ,8)=A2*M1*A(KK,1)**2+A1*M1*A(KK,2)
      A(JJ,9)=A1*M2*A(KK,1)
C
C***** COLLECTOR-BASE CAPACITANCE
C
      NADD=NADD+1
      BR(NADD)=NADD
      NFROM(NADD)=NCJ
      NTO(NADD)=N1
      IF (ABS(C3).EQ.0.0000000) GO TO 105
      TYPE(NADD)=C
      VALUE(NADD)=C3
      KEY(NADD)=2
      NCAP=NCAP+1
      GO TO 110
  105 VALUE(NADD)=1.000E+06
      TYPE(NADD)=R
      KEY(NADD)=5
      NRES=NRES+1
C
C***** EMITTER CAPACITOR(LINEAR)
C
  110 NADD=NADD+1
      BR(NADD)=NADD
      NFROM(NADD)=NJ
      NTO(NADD)=NE
      TYPE(NADD)=C
      VALUE(NADD)=CJE+IE*CP2
      KEY(NADD)=2
      NCAP=NCAP+1
C
C***** BASE-EMITTER CAPACITANCE(LINEAR)
C
      IF (ABS(C1).EQ.0.000) GO TO 115
      NADD=NADD+1
      BR(NADD)=NADD
      NFROM(NADD)=NJ
      NTO(NADD)=NE
      TYPE(NADD)=C
      VALUE(NADD)=C1
      KEY(NADD)=2
      NCAP=NCAP+1
C
C***** COLLECTOR CAPACITIVE NONLINEARITY
C
  115 LL=KK+2
      NADD=NADD+1
      BR(NADD)=NADD
      NFROM(NADD)=NCJ
      NTO(NADD)=NJ
      TYPE(NADD)=NC
      NLBN(LL)=NADD
      A(LL,1)=K/VCB**MU
      A(LL,2)=-A(LL,1)/VCB/6.000
      A(LL,3)=A(LL,1)/VCB**2/27.00
C
C***** COLLECTOR RESISTANCE(LINEAR)
C
```

```
      NADD=NADD+1
      IR(NADD)=NADD
      NFROM(NADD)=NCJ
      NTO(NADD)=NJ
      TYPE(NADD)=R
      VALUE(NADD)=RC
      KEY(NADD)=5
      NRES=NRES+1
C
C***** BASE RESISTANCE(LINEAR)
C
      IR(NEG)=NEG
      NFROM(NEG)=N1
      NTO(NEG)=NJ
      TYPE(NEG)=R
      VALUE(NEG)=RB
      KEY(NEG)=5
      NRES=NRES+1
C
      KK=KK+2
      NNODE=MAX0(NNODE,NE)
C
C***** WRITE TRANSISTOR PARAMETERS
C
      WRITE (6,125) N,VCB,VCDD,MU
      WRITE (6,130) IC,ICMAX,AP,HFEMAX
      WRITE (6,135) K,REF,CJE,CP2
      WRITE (6,140) RB,RC,C1,C3
      RETURN
C
  120 FORMAT (8E10.3)
  125 FORMAT (1H0,22HTRANSISTOR PARAMETERS:,/1H0,2HN=,F7.3,11X,4HVCB=,F7
     1.3,5X,5HVCDD=,F7.3,8X,3HMU=,F8.3)
  130 FORMAT (1H ,3HIC=,E10.3,7X,6HICMAX=,E10.3,4X,2HA=,F7.3,11X,7HHFEMA
     1X=,F7.2)
  135 FORMAT (1H ,2HK=,E12.3,5X,4HREF=,F7.2,9X,4HCJE=,E12.3,4X,4HCP2=,E1
     12.3)
  140 FORMAT (1H ,3HRB=,F8.3,9X,3HRC=,E12.3,5X,3HC1=,E12.3,5X,3HC3=,E12.
     13,/1H0)
C
      END
```

5-4. System Dependent Cards

Program PRANC was developed on the CDC 6500/6600 computer system at Purdue University. The system dependent cards contained in the program are listed in Table 5-1.

Table 5-1. System Dependent Cards

Sub-Program	Card Identification Number
AMAIN	AMN 2200,6000,6430
GZOC	GZC 380
JSEP	JSP 230,250

The sub-programs, and their functions called by the cards listed in Table 5-1 are as follows:

SECOND: Subroutine SECOND is used to determine the elapsed time in seconds in performing a sequence of PRANC phrases.

LINEQ4: Subroutine LINEQ4 is a linear equation solver routine, used to invert a complex matrix.

ISHFTLA (I,N): is used to perform an N-place arithmetic left shift on I (circular).

ISHFTRA (I,N): is used to perform an N-place arithmetic right shift on I (end-off, sign fill); e.g. K = ISHFTRA (1,1) sets K to 0; K = ISHFTRA (1,0) sets K to 1.

CHAPTER 6

CONCLUDING REMARKS

As stated earlier, the fundamental objective underlying this research effort was to examine the computational aspect of the Volterra series method. In the process, we developed an efficient algorithm for adapting the Volterra series method for computer-aided analysis of nonlinear circuits. A semi-symbolic approach for analyzing the linearized part of the nonlinear circuit was used as the basis for this development. The algorithm was implemented in a computer program, entitled PRANC. The main contributions of this effort may thus be identified as follows:

(1) The development of an efficient algorithm for adapting the Volterra series method for computer-aided analysis.

(2) The development of a symbolic approach for analyzing the linearized circuit.

(3) The development of a digital computer program for the spectrum analysis of nonlinear circuits.

As part of the effort, several network examples were exercised on PRANC. The execution times involved in these examples indicate that PRANC is highly efficient from a computational standpoint. Networks with several nonlinearities, several energy storage elements (as in Example 4-2), and multiple input frequencies involve execution times which are small and easily affordable.

The fundamental criterion in the development of PRANC was computational efficiency. The results from the use of PRANC indicate that this criterion has been met successfully. The "ease of use", which is another important performance measure in software development, was not given as much weight in this effort. As part of continuing work, it is recommended that several user-oriented features, such as free-format input, built-in device modelling, parameter variation feature, etc., be incorporated in the program. The computational efficiency inherent in the present version of PRANC together with certain "ease of use" features should render it a powerful tool for analyzing nonlinear circuits.

REFERENCES

[1] Volterra, V., Theory of Functionals and of Integral and Integro-Differential Equations, New York: Dover, 1959.

[2] Wiener, N., Nonlinear Problems in Random Theory, New York: Technology Press, 1958.

[3] Zames, Z. D., "Nonlinear Operators for System Analysis," MIT Research Lab. of Electronics, Tech. Rept. 370, Aug. 25, 1960.

[4] Brilliant, M. B., "Theory of the Analysis of nonlinear systems," MIT Research Lab. of Electronics, Tech. Rept. 345, March 3, 1958.

[5] George, D. A., "Continuous Nonlinear Systems," MIT Research Lab. of Electronics, Tech. Rept. 355, July 24, 1959.

[6] Barrett, J. F., "The Use of Functionals in the Analysis of Nonlinear Physical Systems," Part-I, Sect. 2, A. T. Fuller, ed., Nonlinear Stochastic Control Systems, London: Taylor and Francis, 1970.

[7] Narayanan, S., "Transistor Distortion Analysis Using Volterra Series Representation," BSTJ, 46, pp. 991-1023, May-June 1967.

[8] Van Trees, H., Synthesis of Optimum Nonlinear Control, Mass.: MIT Press, 1962.

[9] Van Trees, H., "Functional Techniques for the Analysis of the Nonlinear Behavior of Phase Locked Loops," Proc. IEEE, Vol. 52, No. 8, pp. 894-911, Aug. 1964.

[10] Graham, J. and Ehrman, L., eds., Nonlinear System Modeling and Analysis with Application to Communication Receivers, Rome Air Development Center, RADC-TR-73-178 (AD-766278), Griffis AFB, June 1973.

[11] Bedrosian, E. and Rice, S., "The Output Properties of Volterra Systems (Nonlinear Systems with Memory) Driven by Harmonic and Gaussian Inputs," Proc. IEEE, Vol. 59, No. 12, pp. 1688-1707, Dec. 1971.

[12] Bussgang, J. J., et. al., "Analysis of Nonlinear Systems with Multiple Inputs," Proc. IEEE, Vol. 62, No. 8, pp. 1088-1119, Aug. 1974.

[13] Thapar, H. K. and B. J. Leon, "Lumped Nonlinear System Analysis with Volterra Series," School of Electrical Engineering, Purdue University, TR-EE-32, July 1979.

[14] Leon, B. J., and Schaefer, D. J., "Volterra Series and Picard Iteration for Nonlinear Circuits and Systems," IEEE Trans., CAS-25, No. 9, pp. 789-793, Sept. 1978.

[15] Lubbock, J. K., and Bansal, V. S., "Multidimensional Laplace Transforms for Solution of Nonlinear Equations," in Multidimensional Systems: Theory and Applications, N. K. Bose, ed., IEEE Press, pp. 155-162, 1979.

[16] Down, T., "Symbolic Evaluation of Transmittances from the Nodal Admittance Matrix," Elec. Letters, Vol. 5, No. 16, pp. 379-380, Aug. 1969.

[17] -, "Inversion of the Nodal-Admittance Matrix in Symbolic Form," Elect. Letters, Vol. 6, No. 3, pp. 74-76, Feb. 1970.

[18] -, "Inversion of the Nodal Admittance Matrix for Active Networks in Symbolic Form," Elect. Letters, Vol. 6, No. 22, pp. 690-691, Oct. 1970.

[19] Lin, P. M., "Formulation of Hybrid Matrices for Linear Multiports Containing Controlled Sources," IEEE Trans. Vol. CT-21, pp. 169-175, March 1974.

[20] Chua, L. O., and Lin, P. M., Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques, Englewood Cliffs, NJ: Prentice-Hall, 1975.

[21] Dahlquist, G., Bjork, A., Numerical Methods, NJ: Prentice Hall, 1974.

[22] Feller, W., An Introduction to Probability Theory and Its Applications, John Wiley and Sons, NY, 1957.

[23] Thapar, H. K. and Leon, B. J., "Computer-Aided Distortion and Spectrum Analysis of Nonlinear Circuits," IEEE Int'l. Symposium on CAS, Tokyo, pp. 46-49, July 1979.

[24] "NCAP: Nonlinear Circuit Analysis Program," Rome Air Development Center, Griffis AFB, NYC, Private Communication.

[25] Lin, P. M., "A Survey of Applications of Symbolic Network Functions," IEEE Trans., Vol. CT-20, pp. 732-737, Nov. 1973.

[26] McClenahan, J., and Chan, S. P., "Computer Analysis of General Linear Networks Using Digraphs," Internl. J. Electronics, No. 22, pp. 153-191, 1972.

[27] Singhal, K., and Vlach, J., "Symbolic Analysis of Analog and Digital Circuits," IEEE Trans., Vol. CAS-24, No. 11, pp. 598-609, Nov. 1977.

[28] Parlett, B. N., "The LU and QR Algorithms," in A. Ralston and H. S. Wilf, eds., Numerical Methods for Digital Computers, NYC: John Wiley and Sons, Inc., 1967, Vol. II, pp. 116-130.

[29] Acton, F. S., Numerical Methods that Work, Harper & Row, Publishers, NYC, 1970.

Appendix A. A DEVICE MODELLING EXAMPLE

In this section we present an example of how to obtain mathematical models for nonlinear devices. The mathematical models so developed can then be used to obtain equivalent circuits for analysis purposes.

Most devices commonly encountered in electronics, where one would be interested in computing the harmonic distortion due to the nonlinear operation, are operated in the active region where the device operation is quasi-linear about an operating point established by the circuit bias. Here we develop the incremental model for some such devices. It is important to make a distinction between total and incremental nonlinear circuits. Total model, or global models, interrelate the total instantaneous voltages, current, and/or charges in the device. Such models are used for operating point or large-signal analysis. The incremental or small-signal models for devices are derived from these global models by some kind of an approximation (usually a Taylor series expansion) around the operating point. In deriving incremental models, it is desirable to have a model that is independent of the bias point in the normal active region, so that the nonlinear effects due to a change in the operating point can be predicted.

We now present a mathematical model for a semiconductor diode. In the commonly used small-signal applications of semiconductor diodes, two types of operations are encountered: (1) forward-bias (e.g. mixers); (2) reverse-bias (varactor converter).

In the forward-bias operation, the primary nonlinearity is a memoryless nonlinearity given by

$$I = I_s[\exp(qV/nkT) - 1] \qquad (1)$$

where n is the ideality factor for the diode. Then for the forward-biased diode with a small-signal input, we can write eqn. (1) as:

$$I_D + i_d = I_s[\exp(qV_D/nkt)\exp(qv_d/nkT) - 1] \tag{2}$$

where I_D and V_D are the bias current and voltage, respectively, and i_d and v_d are the incremental current and voltage. For $(qv_d/nkT) < 1$, we have a convergent Taylor series for:

$$\exp(qv_d/nkT) = \sum_{s=0}^{\infty} \frac{1}{s!}\left(\frac{qv_d}{nkt}\right)^s \tag{3}$$

Substituting (3) into (2) and approximating

$$I_D = I_s\left[\exp\frac{qV_D}{nkT} - 1\right] \approx I_s \exp\frac{qV_D}{nkT},$$

we obtain the following:

$$i_D = I_D \frac{q}{nkT} v_d + \frac{I_D}{2!}\frac{q}{nkT}^2 v_d^2 + \frac{I_D}{3!}\frac{q}{nkT}^3 v_d^3 + \ldots \tag{4}$$

which is in a form suitable for analysis on PRANC.

In the case of the reverse-biased diode, the primary nonlinearity is the nonlinear junction capacitance C(V), where C(V) is of the following form:

$$C(V) = \frac{C(0)}{[1 - V/\phi]^k} \tag{5}$$

where C(0), ϕ, and k are generally specified by the manufacturer.

The charge stored in the capacitor of eqn. (5) is:

$$Q(V) = \int_0^V C(v)\, dv$$
$$= \frac{\phi}{(k-1)} \frac{C(0)}{[1 - V/\phi]^{(k-1)}} \qquad (6)$$

Expanding eqn. (6) into a Taylor series yields:

$$Q(V_c + v_c) = C(V_c) \left[\frac{(\phi - V_c)}{(k-1)} + v_c + \frac{k v_c^2}{2!(\phi - V_c)} + \frac{k(k+1) v_c^3}{3!(\phi - V_c)^2} + \ldots \right] \qquad (7)$$

The incremental capacitor current is the change of total charge with respect to time. Since V_c is a constant, we get

$$i_c = \frac{dQ}{dt} = C(V_c) \frac{dv_c}{dt} + \frac{kC(V_c)}{2!(\phi - V_c)} \frac{dv_c^2}{dt} + \frac{k(k+1)C(V_c)}{3!(\phi - V_c)^2} \frac{d}{dt} v_c^3 + \ldots \qquad (8)$$

Equation (8) is the mathematical model of the incremental nonlinear capacitance current. The first term is a linear capacitor of value $C(V_c)$, and the term in v_c^n represent the nonlinear capacitive terms. Again, note that eqn. (8) is in a form suitable for analysis on PRANC.

The models for other nonlinear devices, such as transistors, JFETS, vacuum tubes, etc., can be found using the same kind of an approach.

MISSION
of
Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control Communications and Intelligence (C^3I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.